Fredrik Sannholm

Automated Treatment Planning in Magnetic Resonance guided High Intensity Focused Ultrasound

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 18.11.2011

Assignment supervisor:

Prof. Raimo Sepponen

Thesis instructor:

D.Sc. (Tech.) Julius Koskela

Aalto University
School of Electrical Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Fredrik Sannholm

Title: Automated Treatment Planning in Magnetic Resonance guided High Intensity Focused Ultrasound

Date: 18.11.2011 Language: English Number of pages:11+95

Department of Electronics

Professorship: Applied electronics Code: S-66

Supervisor: Prof. Raimo Sepponen

Instructor: D.Sc. (Tech.) Julius Koskela

Magnetic Resonance guided High Intensity Focused Ultrasound (MR-HIFU) is a non-invasive medical procedure for localized tissue heating, used mostly in treatment of tumours. The modality utilizes focused ultrasound to raise the temperature of the tumour tissue in small localized volumes, resulting in necrosis. To ablate the whole tumour, several of these *sonication cells* are need. Planning the positions of the cells, while taking into consideration all safety aspects of the treatment, is a time consuming and monotonous task, but requires at the same time expertise and precision. Furthermore, due to the complex characteristics of a MR-HIFU treatment, it is difficult to optimize manually.
The aim of the thesis was to design an outline for an automated treatment planning algorithm for MR-HIFU, and to produce a prototype of such an algorithm. The presented algorithm relies on a step-wise process. First, a set of positions is produced that can be sonicated safely. Then, an optimal subset of those positions is selected. Finally, the remaining treatment parameters are optimized. The treatment can either be optimized for maximum coverage or minimum total treatment time. The proposed algorithm is general enough to be adaptable to all ablation applications of MR-HIFU. It has a modular structure for easy updating, and it is able to improve on the plan during the treatment based on feedback from already delivered cells. This is the first published treatment planning algorithm for MR-HIFU that optimizes the treatment and has the ability to update the plan based on feedback. The prototype was tested in two artificial test cases and one real clinical case, proving its feasibility.

Keywords: HIFU, MR-HIFU, treatment planning, treatment optimization

AALTO-UNIVERSITETETS
HÖGSKOLA FÖR ELEKTROTEKNIK

DIPLOMARBETETS
ABSTRAKT

Författare: Fredrik Sannholm

Arbetets titel: Automatiserad behandlingsplanering inom högintensivt fokuserat ultraljud guidat av magnetresonanstermometri

Datum: 18.11.2011 Språk: Engelska Sidmängd:11+95

Institutionen för elektronik

Professur: Tillämpad elektronik Kod: S-66

Övervakare: Prof. Raimo Sepponen

Handledare: TkD Julius Koskela

Högintensivt fokuserat ultraljud guidat av magnetresonanstermometri (MR-HIFU) är en ickeinvasiv medicinsk metod för att åstadkomma lokal uppvärmning i vävnad, vilket tillämpas främst för behandling av tumörer. Tekniken utnyttjar fokuserat ultraljud för att lokalt höja temperaturen i tumörvävnaden vilket resulterar nekros. För att orsaka ablation i hela tumören krävs det att flera av dessa *celler* sonikeras. Att manuellt planera hur dessa celler skall placeras, medan behandlingens samtliga säkerhetsaspekter tas i beaktande, är en tidskrävande och monoton process som samtidigt kräver expertis och precision. Dessutom, på grund av behandlingens mångfacetterade karaktär är den svår att optimera manuellt.

Syftet med detta arbete var att utforma en algoritm för automatisk behandlingsplanering för MR-HIFU för att förbättra arbetsflödet i planeringsprocessen, samt att producera en prototyp av en dylik algoritm. Den presenterade algoritmen är en stegvis process. Först producerar algoritmen en grupp av positioner som kan sonikeras på ett säkert sätt. Därefter finner algoritmen den optimala undergruppen av dessa positioner. Slutligen optimerar algoritmen resten av de relevanta behandlingsparametrarna. Behandlingen kan optimeras antingen genom att maximera volymen som utsätts för ablation eller genom att minimera tiden som behandlingen kräver. Den presenterade algoritmen är tillräckligt generell för att kunna anpassas till samtliga ablationstillämpningar av MR-HIFU. Den har en modulstruktur vilket förenklar uppgradering, och den kan använda information om hur behandlingen framskrider för att reglera och uppdatera planen. Detta är den första publicerade algoritmen för behandlingsplanering inom MR-HIFU som kan optimera behandlingen samt använda återkoppling för att reglera planen. Prototypen testades i två konstgjorda fall samt i ett äkta kliniskt fall vilket dess genomförbarhet.

Nyckelord: HIFU, MR-HIFU, behandlingsplannering, behandlingsoptimering

Förord

Detta diplomarbete har utförts hos Philips Medical Systems MR Finland. Som övervakare för arbetet fungerade professor Raimo Sepponen och som handledare teknologiedoktor Julius Koskela.

Jag vill börja med att tacka professor Sepponen för handledning och granskning av arbetet. Jag vill dessutom lyfta fram Julius insatts som handledare. Han har inte bara handlett och gjort sitt bästa för att sidmängden i arbetet skulle hållas inom rimliga gränser, utan även sett till att jag har kunnat få den tid och de resurser som krävts för kunna slutföra arbetet. Stort tack, Julius!

Utöver detta vill jag speciellt tacka mina kolleger dr Charles Mougenot, dr Jukka Tanttu, Pirjo Wirtanen samt Mika Yli-Hautala för deras insats och den sakkunnighet som de bridragit med i genomförandet av detta arbete. Ett tack går även till Jyrki Lötjönen, VTT, som har bidragit med sitt tekniskt kunnande. Jag vill dessutom passa på att tacka mina övriga kolleger på Philips för all hjälp och det intresse de visat gentemot mitt arbete. Det har varit ytterst motiverande att jobba tillsammans med er!

Därutöver vill jag tacka mina studiekompisar som visat mig vad det betyder att vara teknolog. Ni har gjort min studietid oförglömlig!

Jag vill dessutom tacka min familj: mamma, pappa och Filip. Alla har ni stött och trott på mig under hela min studietid, vilket jag är ytterst tacksam för! Ett tack går även till Piipa, Katja, Silja och hela *Routionväki* som aldrig slutade kämpa på mig. Slutligen vill jag lyfta fram min sambo Michaela. Hon har inte bara stått ut med mig utan under hela den här processen utan även varit min stöttesten och tillflykt. Tack för allting, Mixo!

Esbo, 18.11.2011

Fredrik Sannholm

Contents

Symbols and abbreviations

Abbreviations

ANN Artificial Neural Networks

AP Assignment Problem

ATA Available Treatment Area

BB Branch-and-Bound algorithms

BC Branch-and-Cut algorithms

BHE Pennes' Bioheat Transfer Equation

BMC Budgeted Coverage Maximum problem

BUC Budgeted Unique Coverage problem

EM Equivalent Minutes at 43^{o}C, the unit of thermal damage

FF Far Field

FN Farthest Neighbour algorithm for LPP

GA Genetic Algorithms

GLPK GNU Linear Programming Kit

GRASP Greedy Randomized Adaptive Search algorithm

HDR High Dose Rate brachytherapy

HIFU High Intensity Focused Ultrasound

IP Integer Programming

JD Judgement Day operator

LP Linear Programming

LPP Longest (Hamiltonian) Path Problem

LS Local Search method

MCCT Minimum Cost Coverage Threshold problem

MCUCT Minimum Cost Unique Cover Threshold problem

MR Magnetic Resonance

MR-HIFU Magnetic Resonance guided High Intensity Focused Ultrasound

MRI Magnetic Resonance Imaging

NF Near Field

NN Nearest Neighbour algorithm for TSP

OAR Organ(s) At Risk

PRF Proton Resonance Frequency

PTV Primary Target Volume, or Planned Target Volume

RF Radio Frequency

ROI Region of Interest

SA Simulated Annealing algorithms

SUS Stochastic Universal Sampling

SW Software

TP Target Points

TSP Travelling Salesman Problem

UF Uterine Fibroid

Symbols and Operators

α Opening angle of a cone

α_{a} Absorption coefficient

β Decision parameter for the GRASP algorithm

ϕ Angle between the major axis of an ellipse and the x-axis

ρ Tissue density

$\sigma(t)$ Standard deviation of the fitness values in a population, at generation t

θ Parameter used to define an elliptic curve

$\bar{A}$ Axis of a cone

A_{ij} Arc between node i and j

B Budget

$\mathbf{C}$ Coverage matrix

c_i Cost of cell i

c_{b} Specific heat of blood

c_{t} Specific heat of tissue

c_{ij} The cost or distance of the arc between node i and j

$\mathbf{D}$ Distance matrix

d Distance

D_{ij} Entry (i, j) in the distance matrix $\mathbf{D}$

$\bar{E}_0$, $\bar{E}_1$ Vectors that define a plane

$ExpVal(i, t)$ Expectation value of chromosome i, at generation t

$f(i, t)$ Fitness value of chromosome i, at generation t

$\bar{f}(t)$ Mean fitness value in a population, at generation t

$I(\bar{r}, t)$ Intensity distribution

k_{t} Heat conductivity of tissue

$\bar{l}_{ij}$ Line between the centres of two ellipses

M Number of cells

N Number of (target) points, or nodes

$\bar{P}_0$, $\bar{P}_1$, $\bar{P}_1$ Vectors representing points on a plane

$\bar{P}_{\mathrm{c}}$ Centre point of an ellipse

$\bar{P}$ Point on an elliptic curve

$Q(\bar{r}, t)$ Heat distribution

r Parameter used to define a point on a plane

R_{ij} The radius of ellipse i towards the centre of ellipse j

$\mathcal{S}$ Family of sets or cells

$\mathcal{S}'$ Subfamily of sets or cells

s Parameter used to define a point on a plane

$\Delta t_{\mathrm{cool,i}}$ Cooling time for sonication i

$\Delta t_{\mathrm{heat,i}}$ Heating time for sonication i

TD_{43}	Thermal Damage
T	Treshold
t	Time
$T(\bar{r}, t)$	Temperature distribution
$T(\tau)$	Time-dependent temperature
T_{ar}	Temperature of the arterial blood
T_{t}	Total treatment time
$\bar{V}$	Vertex of a cone
v, w	Length of major and minor axis of an ellipse
w_j	Weight of (target) point j
w_{b}	Perfusion rate of blood
$\bar{X}$	Point on a plane
$\bar{x}_{\text{p}}$	Calculation point
$\bar{x}_{\text{s}}$	Seed position
$\mathcal{X}$	Set of (target) points
$x_{\text{a},ij}$	Boolean indicating if the arc between node i and j has been chosen
x_j	Boolean indicating if (target) point j is covered
x_{ij}	Boolean indicating if (target) point j is covered by cell i
y_i	Boolean indicating if cell i is chosen

1 Introduction

Magnetic Resonance guided High Intensity Focused Ultrasound (MR-HIFU) is a non-invasive medical procedure for localized tissue heating, utilized mostly in treatment of tumours. The modality is a hybrid combining the therapeutic abilities of HIFU and the imaging capabilities of *magnetic resonance imaging* (MRI). The technique utilizes concave transducers to produce a focused ultrasound beam. The beam concentrates the ultrasound energy in a single spot in the targeted volume. The deposited energy causes local heat accumulation, which in turns leads to temperature rise and eventually necrosis. The procedure is guided using MR-thermometry, allowing real-time *in vivo* monitoring of the temperature development during the delivery of the treatment. Even though the use of ultrasound for hypertherapy is well known, combining the technique with MR-thermometry is a relatively new development. The superior soft-tissue contrast, high resolution, and the ability to measure relative temperature changes imply that MRI not only allows the guiding of the treatment, but also facilitates monitoring the safety of the delivery as well as the planning of the treatment and the verification of the results. A further, promising application for MR-HIFU is its use in drug-delivery. The drugs are contained inside thermosensitive capsules that release the drug locally when heated, effectively limiting the spread of the drug to the rest of the body.

The Philips Sonalleve MR-HIFU system is one of the few commercially available MR-HIFU solutions. The system incorporates a specially designed table top, which contains the transducer, and a software console used to plan, control, and monitor the treatment. The main application for the Sonalleve MR-HIFU system is the treatment of uterine fibroids (UF), benign tumours located in the uterus. Recently, first treatments of bone metastasis using the systems have been performed, and new applications for tumours in other anatomical regions, such as the prostate and breasts, are in development.

As the population in the world is ageing and continuing to grow, the number of new cancer cases per year continues to increase. According to statistics published in [1], more than one third of the population (in the UK) will develop some form of cancer during their lifetime. Breast and prostate cancer are among the most common types of cancer, with approximately 1.3 and 0.9 million new incidences per year worldwide, respectively. Breast cancer is the most common form of cancer among women, while prostate is the second most common among men. [2] Both are also potential future applications for MR-HIFU treatments. As mentioned, the main application of MR-HIFU has been the treatment of uterine fibroids. Despite not being malignant, UFs reduce the life quality of a large portion of the women in the world. It has been reported that up to 25% of women in child bearing age shows symptoms of uterine fibroids. These include heavy and prolonged menstrual bleeding, severe pain, bloating, constipation and/or urinary complaints. [3]

Traditional, surgical methods of treating UF include *hysterectomy*, *myomectomy*, and *embolization*, all of which are invasive, require some level of hospitalization and recovery time, and involve risks for the patient. More generally, non-surgical methods for pathology treatment include ablating the tumour by radiation and using temperature based therapies. Radiation-based techniques, such as *brachytherapy*, destroy the tumour tissue by utilizing ionizing radiation. The modalities are relatively old and also expose healthy tissue to damaging radiation and brachytherapy, in particular, has the further disadvantage of being

invasive. Heating-based methods, such as radio-frequency, microwave or laser ablation, are also invasive. [4] *High intensity focused ultrasound* (HIFU), on the other hand, is non-invasive, minimizing the risk of bleeding and infections, and also does not utilize any kind of ionizing radiation [5]. The recovery time following a MR-HIFU treatment is usually very short, and the patient may leave the hospital the same day.

The term *treatment planning* incorporates the process of planning the delivery of the treatment on the console software, by defining the treatment parameters that influence the outcome of the treatment and by taking into consideration the safety of the patient as well as the physical constraints of the equipment. Often, an optimization aspect is also included in the process. On the Phillips Sonalleve MR-HIFU system, the treatment planning process is currently performed manually, which can be a time consuming and monotonous task, considering the number of cells required to treat a large tumour and the extensive requirements that each treatment cell should fulfil. Furthermore, the complexity of a treatment delivery, as a system, and the number of variables that influence the outcome, make optimizing the treatment manually an overwhelming task. Only a few published reports can be found on the subject of treatment planning in the HIFU environment [6]. In radiotherapy and brachytherapy, automated treatment planning algorithms have long been the subject of development, and highly sophisticated commercial solutions are available. The algorithms conduct so called *inverse planning* as they produce optimal sets of treatment parameters based on the user defined target areas. This is in contrast with *forward planning*, which attempts to optimize the parameters through trial-and-error.

The aim of this thesis is to produce an outline for an automated treatment planning algorithm for the Phillips Sonalleve MR-HIFU system, and to produce a functional prototype of the algorithm, as well as to introduce and compare a number of alternative methods of implementation. The primary aim of the algorithm is to improve the planning workflow so that the users prefer to use the algorithm over manual planning. The algorithm should be able to perform an inverse planning procedure, taking as input the target volume and other treatment constraints, and producing as output a complete, optimized treatment plan that fulfils all safety requirements and takes equipment limitations into account. Moreover, seen from the point of view of the user, the algorithm should be easy to use, but also versatile enough to handle and adapt to varied treatment cases. The outline of the algorithm should be general enough to function for a wide range of treatment applications, but within this thesis the main focus will be on the uterine fibroid application.

The first part of this thesis is a literary review, introducing the basics behind HIFU and MR-HIFU, its applications, the treatment parameters affecting the outcome of the treatment, and prior studies on the optimization of those parameters. As publications on HIFU treatment planning are scarce, the optimization aspect of the planning algorithm is approached by presenting two similar applications, which have received more attention in literature: brachytherapy and network planning. The second part of the thesis discusses the requirements for a treatment planning algorithm in the Sonalleve environment in further detail, before the proposed outline of the treatment planning algorithm is presented. In the third part, the implemented algorithm prototype is introduced. The performance of the optimization algorithms is compared in artificial and real clinical test cases. The last part contains a summary of the thesis.

Part I
Literary Review

2 Introduction to HIFU

The first comprehensive studies related to the biophysical effects of *ultrasound* were conducted in 1927 by Wood and Loomis, who reported that high intensity ultrasound had the ability to rapture cell membranes in its propagation path. The possible applications of this phenomenon were quickly realised, and in 1933 Szent-Györgi was the first to propose the use of ultrasound for the treatment of cancer. Further investigation was conducted into the possible practical application of high-intensity ultrasound for some decades. Lynn *et al.* introduced in 1942 the concave transducer, which focused the ultrasound waves to a single focal point and thus increased the local intensity of the waves while sparing the tissue around the point. This development can be considered as the beginning of HIFU as a technique [7, 8]. The first applications for use in humans, for the treatment of e.g. Parkinson's disease, were reported in 1960's [8]. Even though the technique showed potential, the most substantial challenge regarding the clinical use of ultrasound for tissue ablation was the lack of guidance and thus the difficulty of targeting the beam [7].

In order to solve the problem, several different imaging modalities have been proposed to be used for guiding the HIFU treatment. The most intuitive modality is probably *ultrasonography*, i.e. the use of ultrasound to image subcutaneous anatomical structures. Due to technical limitations the image quality of the modality is relatively low, which renders the guiding of HIFU more challenging. In spite of this, there exist today commercial HIFU products using ultrasonography as a guiding modality. [6] The use of MRI as guidance for HIFU treatments was first proposed by Cline in 1993. MRI is an intensively studied and well-develop imaging modality which is extensively used for medical diagnostics, and which offers excellent soft tissue contrast as well as high resolution. Moreover, several MR tissue parameters change as a result of temperature increase and tissue damage, which allows MR not only to visualize the induced tissue damage after the treatment but also to produce high resolution *in vivo* volumetric temperature maps within the tissue in real-time. In other words, MRI allows the operator to non-invasively monitor the temperature rise inside the body during the treatment. [7]

This section introduces the physical and physiological phenomena underlying MR-HIFU treatments as well as some of its applications, and gives a description of a typical MR-HIFU treatment workflow using the Philips Sonalleve MR-HIFU platform.

2.1 Ultrasound in Human Tissue

Sound waves oscillating at frequencies above 20 kHz are classified as ultrasound. Ultrasound waves propagate in solid, liquid and gaseous media and thus also in human tissue. As the ultrasound waves travel through tissue, the pressure oscillations cause the cells and molecules in the tissue to vibrate and rotate, which in turn results in a rise in the local temperature. This interaction with molecules causes the ultrasound waves to lose energy.

Furthermore, ultrasound waves propagating through the numerous interfaces between different tissue types at complicated angles in the body experience reflection and refraction continuously. This causes it not only to scatter and lose energy but may also shift or distort the focal point of a focused beam. The decrease in energy, or pressure, is called *attenuation*. The *attenuation coefficient* is tissue dependent, and tends to increase with the frequency [9], as well as with the tissue compactness. Blood has a low coefficient, muscle somewhat higher, and attenuation in bone is two orders of magnitude higher than in blood. The coefficient can, however, vary considerably between individuals, tissue types and even within inhomogeneous tissue (such as uterine fibroids). [7]

Local heating in biological tissue will cause the temperature to rise locally, counteracted by *diffusion* and *perfusion*. Diffusion is the spread of heat from areas with more heat to areas with less. The phenomenon is fairly local. Perfusion, on the other hand, can be quite effective at removing heat from a high concentration area and redistributing it to other parts of the body. Perfusion is a consequence of blood flow. Blood binds heat and transfers it by convection. Consequently, the perfusion rate depends on the rate and the direction of blood flow. The body uses blood flow and perfusion actively to regulate its temperature. The temperature rise in biological tissue is most often described by *Pennes' bioheat transfer equation* (BHE), which is presented in more detail in Section 2.1.1.

Tissue can withstand moderately raised temperatures for some time, but continued exposure to temperatures above the normal causes damage to the tissue. In order to quantify the thermal damage to tissue Sapareto and Dewey introduced the concept of *thermal dose* [7]

$$TD_{43}(t) = \int_0^t R^{(43^o\mathrm{C}-T(\tau))} d\tau, \qquad (2.1)$$
$$\text{where } R = 0.25, \text{ for } T < 43^o\mathrm{C},$$
$$R = 0.5, \text{ for } T \geq 43^o\mathrm{C},$$

where $TD_{43}(t)$ is the thermal dose, t is time and $T(\tau)$ is the time-dependent temperature. Thermal dose is measured in equivalent minutes (EM) at 43^oC, indicating the time that the tissue would need to be kept at 43^oC in order to sustain an equal amount of damage. According to Equation (2.1), the higher the temperature the shorter time is needed to inflict damage on the tissue, and *vice versa*. 240 EM is usually considered to be the threshold for full coagulative necrosis (premature cell death) in the tissue, and 30 EM a predictor for onset of tissue damage, even though these thresholds are somewhat tissue dependent. Tissue damage caused by elevated temperature is called *thermal lesion* or *ablation*, and may present itself as, for example, ruptured cells, denaturized proteins, oedema, and coagulated veins. [7]

2.1.1 Pennes' Bioheat Transfer Equation

Pennes' bioheat transfer equation describes the time evolution of the temperature distribution in biological tissue [7]:

$$\rho c_{\mathrm{t}} \frac{\partial T(\bar{r}, t)}{\partial t} = k_{\mathrm{t}} \nabla^2 T(\bar{r}, t) - w_{\mathrm{b}} c_{\mathrm{b}} (T(\bar{r}, t) - T_{\mathrm{ar}}) + Q(\bar{r}, t). \qquad (2.2)$$

Here, $T(\bar{r}, t)$ is the temperature distribution, ρ is the density of the tissue, c_t and c_b are the specific heat of the tissue and the blood, respectively, k_t is the heat conductivity of the tissue, w_b is the perfusion rate of the blood, T_ar is the temperature of the arterial blood, and $Q(\bar{r}, t)$ is the heat introduced into the system. In the case that the heat is caused by absorbed ultrasound waves, the heat term is $Q(\bar{r}, t) = 2\alpha_\mathrm{a} I(\bar{r}, t)$, where α_a is the absorption coefficient of the ultrasound waves and $I(\bar{r}, t)$ is the intensity of the waves. The first term on the right hand side of Equation (2.2) describes heat diffusion, while the second term describes perfusion. As can be seen, perfusion depends on the temperature difference between the tissue and the arterial blood. A partial differential equation such as Equation (2.2) is extremely difficult to solve analytically for relevant boundary conditions and tissue geometries without extensive simplifications. Therefore, various numerical methods are generally applied to solve the equation. [10]

2.2 HIFU

The ultrasound used in HIFU applications is produced by several small piezoelectric transducer elements arranged in the shape of a spherical shell [7, 8]. The array of small elements creates together, through constructive interference, a conical beam with a focus point close to the geometrical centre of the transducers curvature. This is visualized in Figure 1. The intensity in the focal point can be three orders of magnitude larger than the intensity of one plane wave produced by a single element, and typically several orders of magnitude larger than those used in ultrasonography. The transducers usually operate at frequencies in the range 0.8-4 MHz, depending on their use, the sonication depth, and the target. [7]

The process of delivering ultrasound to tissue with the intent of causing ablation is called *sonication*, and the constrained volume of tissue that is to be ablated during a single sonication event is called a *treatment cell* or simply a *cell*. The focal spot has in an ideal case the form of an ellipsoid, the size of which depends on the transducer geometry, the speed of sound in tissue, and the frequency of the waves. [7, 8, 11] Differences in tissue parameters and interfaces between tissue types affect the propagation of the ultrasound, which influences the form of the focal spot. Especially bone tissue, with a relatively high absorption coefficient, causes large portions of the ultrasound energy to be absorbed, which in turn may cause unintended heat distributions. [12] For this reason, care is taken to avoid bone in normal uterine fibroid treatments with HIFU.

In order for the transducer to ablate at different positions, the focal point of the ultrasound cone needs to be moved. This can either be done mechanically or electronically. Physically moving the transducer around during the treatment induces, however, significant artefacts in the MRI temperature monitoring. Electronically controlling the position of the focal point requires the use of so called *phased-array transducers*. In a phased-array transducer, the small elements can be individually controlled, allowing the operator to set the phase and amplitude of each element separately. By altering the phase of the individual waves, it is possible to accurately control where the constructive interference creates the focal spot. [7, 8] This technique is called *electronic steering* and the physical phenomenon is called *electronic deflection*. [7] The phenomenon is demonstrated in Figure 1

As the size of the targeted tumours are much larger than the focal point, the translation range that the electronic steering provides is usually not enough to allow ablation of

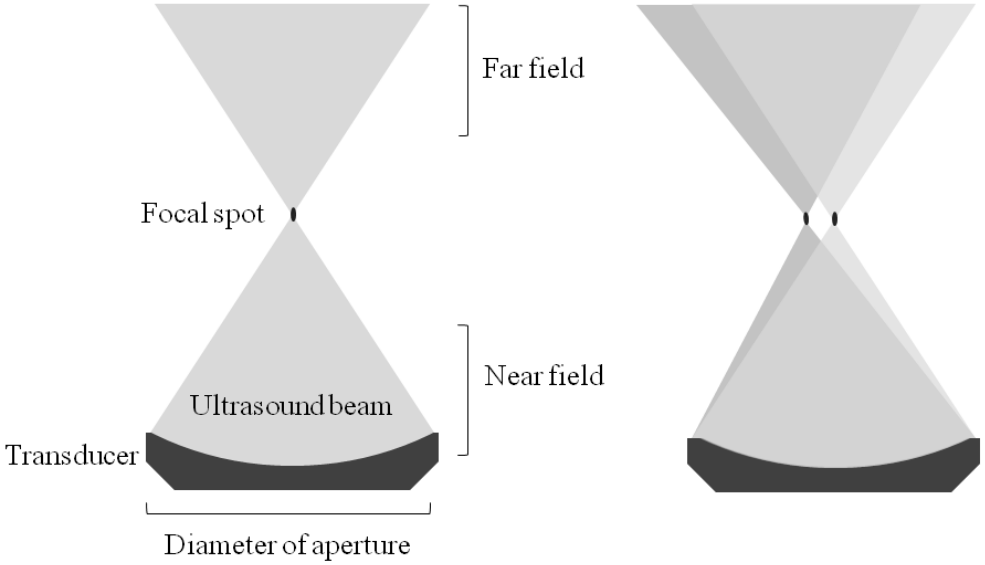


Figure 1: The left figure visualizes and labels important parts of the ultrasound beam produced by the transducer. The right figure demonstrates the effect of electronic steering on the ultrasound beam.

the whole target without mechanically moving the transducer between sonications. Moreover, the focal point needs to be positioned very accurately. For these reasons modern HIFU-devices typically employ some sort of automation to handle the movement of the transducer with sufficient precision. The electronic steering is instead used to increase the efficiency of the sonication.

Traditionally, sonications are performed by ablating a single focal point at a time, moving the transducer slightly while letting the near field (NF, see Figure 1), containing fat, skin and muscle tissue, to cool down before sonicating again. The volume of necrosed tissue depends on the power of the ultrasound waves, the duration of the sonication [8, 11], and tissue parameters [8]. Typically, heat is deposited quickly enough for the thermal distribution to be largely independent of perfusion effects, and the boundary between healthy and damaged tissue is relatively sharp. [8] Heat build-up from one sonication is typically small, but the cumulative effect due to multiple sonications and inadequate cooling between the sonications may cause tissue damage in the near field [11]. For this reason, cooling times are crucial. The single-sonication method is inefficient as a large part of the heat produced in the preceding sonication is lost during the cooling period, while it could be utilized to raise the temperature in the surrounding tissue more quickly [7, 13]. Furthermore, the cooling times between the sonications prolong the total treatment time [11, 14]. By taking advantage of the rapid and precise steering of the focal point that the phased-array transducer offer, modern HIFU-devices are able to ablate volumes larger than the focal point per sonication by scanning the area around the focal point at precisely controlled speeds and along specified trajectories. This allows for a considerable reduction in treatment times, as several cooling periods can be eliminated, and thus increases the efficiency of the treatment. [7, 13] Numerous studies have been made into the development of these *volumetric*

sonication strategies [7], with the most advanced employing feedback algorithms to optimize the power, duration and trajectory during the sonication to adapt to the unpredictable heating of heterogeneous tissue. [7, 13]

The increase in the sonicated volume also increases the amount of energy deposited in the tissue per sonication. This, in turn, increases the heating in the near field and, consequently, the risk of adverse effects such as skin burns. The main limitation for the use of HIFU for tumour treatment is, in fact, the long treatment times due to the risk of near-field overheating [4, 15]. Although the use of volumetric sonication methods leads to longer cooling times between sonications, it has been shown that the total treatment time is still decreased. The heat build up in the near field can, also, be reduced by minimizing the overlap of successive sonications. [7] All in all, very little research has been performed to finding the optimal balance between sonication durations and cooling times.

2.3 MRI Guidance and MR-HIFU

Magnetic resonance imaging is used extensively in medical diagnostics due to its excellent soft tissue resolution and high versatility. In short, MRI utilizes the inherent magnetization of certain atomic nuclei to image the internal structures of an object. Medical diagnostic applications utilize the magnetization of hydrogen atoms, found in abundance in water and organic molecules. Inside a MRI scanner a strong magnetic field, typically 1.5 T or 3 T, acts on the magnetization of these nuclei and forces the atoms to align themselves in parallel with the field. By applying a short radio frequency (RF) pulse the magnetizations are provided energy that disturbs this equilibrium state. As the nuclei gradually returns to their natural alignment they release the gained energy in the form of oscillating magnetic fields, which can be detected by the scanner. Hydrogen atoms in different chemical environments, such as water and fat, react differently to the external and the RF field and they therefore relax back to the equilibrium state at different paces. The modality is, therefore, able to distinguish between tissue types and create cross-sectional and volumetric representations of the internal structure of the imaged volume. The ability to distinguish between tissue types gives the modality excellent soft-tissue contrast. Applications for MRI are vast, ranging from traditional diagnostic imaging to functional MRI. As MRI does not utilize any ionizing radiation, the safety concerns related to MRI are minimal compared to alternative modality such as Computer Tomography (CT) and Positron Emission Tomography (PET).

The capability of MRI to display contrast between soft-tissue at high resolutions ensures that it is well-suited for locating pathological tissue in healthy tissue, which in turn facilitates the planning of the MR-HIFU treatment [7, 13]. MRI is also used in post-treatment to determine the extent of the induced necrosis. [7] However, without knowing the exact absorption and perfusion constants of and the detailed structure and type of the tissue, through which the ultrasound propagates, it is difficult to predict how the ultrasound will be absorbed and thus how the temperature will rise in the tissue. [7] For safety reasons it is therefore vital to utilize on-line temperature monitoring during the treatment to ensure that no sensitive organs are at risk. [7, 13] MR thermometry methods allow the operator to non-invasively measure the temperature change inside tissue. Temperature elevations are usually monitored specifically in the near field, in the *far field* (FF), along the beam

path and at the target region. A sufficient temporal resolution is used so that the operator can intervene if necessary. Moreover, the temperature maps provided by MR thermometry can be utilized as feedback data to alter and guide the sonications to ensure optimal treatment results. [7]

Several methods for using MRI for temperature imaging have been suggested, all measuring different temperature dependent MR parameters. Changes in temperature cause changes in the way the molecules in tissue interact, altering the chemical environment of the hydrogen nuclei and hence also their MR parameters. The most widely used thermometry technique is based on the changes in *proton resonance frequency* (PRF). [7] A detailed description of PRF thermometry is beyond the scope of this thesis, but it should be noted that the technique is only capable of measuring relative temperature changes and not the absolute temperature. This presents some limitations for MR-HIFU using the technique: the core body temperature is usually used as a starting temperature, on top of which the temperature variation maps from the thermometry scans are added to obtain the absolute temperature maps [7]. If local hotspots exist after a prior sonication, a newly initialised thermometry scan will not be able to communicate the absolute temperature correctly [7, 16].

2.4 Applications

The possible clinical applications of MR-HIFU are numerous and have spurred a lot of interest during the short history of the modality. The applications are, however, constrained by the difficulty of delivering ultrasound through bone and gas filled cavities. As mentioned above, MR-HIFU has primarily been used for treating uterine fibroids. HIFU has proven to have superior specificity compared to other available techniques for hysterectomy, and as it is non-invasive it also has shorter recovery times and the risks of adverse events, such as inflammations, are smaller. Similarly, there have been several studies proving the feasibility of ablation of prostate cancer using HIFU guided either by MRI or ultrasonography. [7]

Other interesting applications of MR-HIFU include ablation of tumours in the breast and bone [3, 7, 12]. The successful ablation of benign breast fibroadenoma and malignant breast carcinoma has been reported, as well as alleviating treatment of bone metastasis [7]. An even more exciting application of focused ultrasound heating is the use for temporally and spatially accurate drug delivery and gene therapy. The drug, for example a chemotherapeutic drug whose spread into the rest of the body should be minimized, is encapsulated in thermosensitive liposomes that release the drug when heated. By increasing the temperature of the tissue locally using HIFU, the drug is released, and its spread can be restrained to the heated volume [17–19]. Furthermore, the disruption of the blood-brain barrier by HIFU has been shown to allow drugs to enter the central nervous system, increasing their effectiveness [20].

Several other applications of HIFU, with or without MR guidance have been reported. These include treatment of tumours in the liver, the kidneys, the bladder, and the eye, as well as stroke treatment and several neurosurgical applications [7, 8, 8, 12]. Early treatments of the brain required that a part of the skull was removed to reach the target [7, 8]. However, recently there have been reports of non-invasive treatments having been carried

out without the removal of parts of the skull [5, 20], thus further decreasing the risk of inflammations and other adverse effects.

Recently, preliminary results presented by Voogt *et al.* [21] have shown promising results for using MR-HIFU for tumour vessel ablation and occlusion, cutting of the blood supply to the tumour. This reduces perfusion, allowing the deposited energy to cause more damage to the tumour. Devascularisation of the tumour has also shown to deprive the tumour of oxygen and nutrition, leading to coagulative necrosis, but also to inflict collateral damage on smaller tumours in the vicinity of the larger, primary target of the treatment, increasing the efficiency of the treatment considerably [22].

2.5 Phillips Sonalleve MR-HIFU Platform

The Phillips Sonalleve MR-HIFU platform consists of a table top (Figure 2), containing the transducer as well as electronics and mechanics to acquire the MR scans and to control the transducer, a generator cabinet, and a console with which the system is controlled. The system is installed adjacent to the MRI scanner, and exchanges considerable amounts of information with the scanner. The table top has the capabilities to function as a normal patient table for MRI scanners, but the HIFU-application sets special requirements on the table. First, in order for the ultrasound to propagate from the transducer to the patient, a transparent window membrane is located approximately in the middle of the table. The part of the patient that is to be treated is positioned carefully over the membrane, ensuring good reach for the ultrasound beam and that sensitive organs can be avoided. Second, as ultrasound cannot travel effectively through air, the transducer is submerged in water or oil to provide an acoustic path for the beam to propagate through. A phased-array transducer is used to allow electrical steering.

The console software allows the operator to import planning images from the scanner and use them in the planning stage of the treatment. The graphical user-interface (Figure 3) also allows the user to plan the positions of the treatment cells, using graphical overlays of the transducer and the beam path to facilitate safety checks. Finally, the software allows the user not only to control the transducer and the power used in each sonication, but also to monitor the development of the heat and dose accumulation on top of low-resolution anatomical scans during the sonication.

2.5.1 MR-HIFU Treatment Workflow

A typical MR-HIFU treatment using the Philips Sonalleve MR-HIFU platform can be divided into three separate phases: the planning phase, the sonication phase, and the post-treatment verification phase [7]. Initially the patient is positioned on the MR-HIFU table top with the target area aligned over the ultrasound window. If the patient moves during the treatment his/her position needs to be corrected during the treatment, between sonications.

The initial part of the planning phase consists of acquiring high-resolution scans of the targeted volume with surrounding tissue and the skin interface. The scans are used to locate the tumour and *Organs at Risk* (OAR), such as the spine and bowel, in the near and far field [3]. In the next step, treatment cells are positioned, or *populated* inside the

tumour so that the ablated tumour volume is maximized, while taking safety issues into consideration. The process of positioning cells inside the tumour is called *population*. As this procedure is carried out manually by the operator and as each cell needs to be checked against a number of safety requirements, this can be a time consuming process. In UF treatments, the safety requirements dictate that the cells cannot be positioned too close to the perimetrium (the serosa of the uterus), the ultrasound beam cannot pass through any OARs in the near field, and there should be a cell-size dependent safety margin between the cell centre and any OARs in the far field. Furthermore, the cells should, in general, not overlap. The cells can be positioned freely inside the target volume in three dimensions, x-direction (anterior-posterior), y-direction (left-right) and z-direction (head-feet), and using two tilt angles, *roll* (rotation about the z-axis) and *pitch* (rotation about the y-axis). In this thesis a *position* is defined in a five dimensional space having x-, y-, z-, *roll*- and *pitch*-coordinates, while a *point* is defined in the Cartesian coordinate system. The physical constraints of the transducer (the focal length, the range of the electrical deflection) and the positioner responsible for the physical movement of it (the size of the oil container, joints' maximum angles, *etc.*) limit the transducers *available treatment area* (ATA) .

There does not exist, to the writer's knowledge, any reported research into the optimal positioning of treatment cells inside the target volume, and at the moment treatments are planned using only the experience and clinical expertise of the operator. However, as the aim of the treatment is to ablate a maximum volume of the tumour, it can be assumed that the operator will try to minimize the gap between adjacent cells. Furthermore, the operator needs to balance the total treatment time, as it is the main limiting factor in HIFU treatments and the patient comfort needs to be considered.

Figure 2: The Philips Sonalleve MR-HIFU table top with a Philips Ingenia 3T MRI scanner. The picture was taken (November, 2011) from the official Philips Sonalleve marketing website, at http://www.healthcare.philips.com/main/products/mri/systems/sonalleve/.

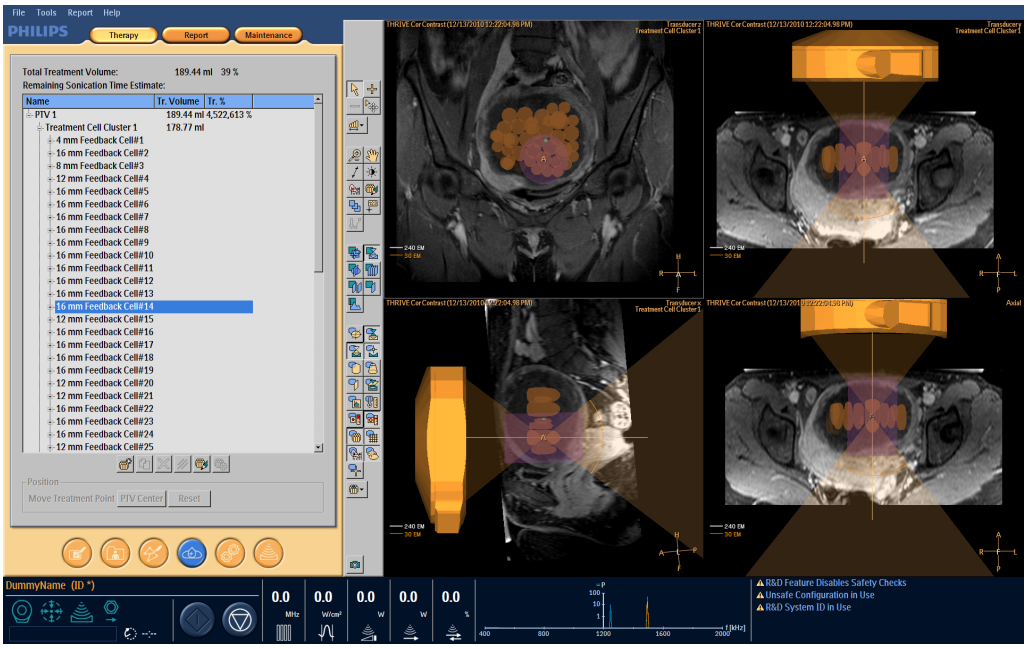


Figure 3: The Phillips Sonalleve MR-HIFU software GUI. On the left side the planned treatment cells are listed. On the right are the planning images, viewed from three different angles, together with a visualization of the planned cells. The transducer and the US beam are displayed as graphical overlays to help the planning process.

The sonication phase of the treatment, in which the actual treatment is performed, starts with carrying out a test sonication at a low power level. Test sonications verify the transducer position and ensure proper ultrasound coupling from the transducer to the cell position. This is an essential step in ensuring safe sonications at proper power levels, as poor coupling in the ultrasound beam path could cause severe burn damage to the patient. [15] Once the coupling has been verified the actual treatment sonications can be conducted. During the sonications, the operator monitors the heat and thermal dose build-up in real time to ensure that the delivery is safe. If the heat build-up develops in a way that puts the patient's safety at risk, the operator aborts the sonications. Moreover, the patient is able to terminate the treatment if the heat causes pain or discomfort. After the sonication, the cooling of the tissue is approximated by analysing the rate at which the temperature in the tissue decreases. This allows the cooling period succeeding the sonication to be minimized. After the cooling period, the next cell to be sonicated is selected by the operator in order to minimize the risk of NF heat accumulation and resulting skin burns. The sonication process is then repeated. The operator is able to add new cells or remove and/or alter already planned cells in between sonications, depending on the outcome of the sonicated cells.

The result of the treatment is verified in the post-treatment phase. These images are usually equally detailed as the planning images, but are optimized to show the change in tissue parameters caused by necrosis in the ablated areas of the tumour. [7]

3 MR-HIFU Treatment Planning and Optimization

In order for a treatment planning algorithm to improve on the planning workflow and produce better treatment plans, it needs an optimization procedure that can balance the treatment parameters in such a way that the outcome of the planned treatment is (approximately) optimal. Moreover, partly due to the young age of the modality and partly due to the complexity of the interrelated factors affecting the outcome of the treatment, relatively few papers have been published on the subject. Moreover, published papers tend to concentrate only on one aspect of the treatment. In this section, a brief review is given over publications dealing with HIFU treatment optimization.

In essence, HIFU treatment optimization is a multi-objective optimization problem, often with conflicting aims. The primary objective is to deliver a safe and effective treatment: to ablate the tumour and to alleviate the pain. The secondary objective is to minimize the total treatment time, for two reasons. First, prolonged treatments are uncomfortable for the patient, who is required to lie very still during the treatment for safety reasons [23]. Second, the time used in a MRI scanner is expensive [11]. Prolonged total treatments times are regarded as the main limitation of MR-HIFU treatments [4]. The total treatment time consists of the time spent positioning the patient on the table top, acquiring the planning images, planning the treatment, together with other pre-treatment procedures, the actual delivery of the treatment, acquiring verification images and other post treatment procedures. The time used to deliver the treatment can be approximated as [11, 24]

$$T_{\mathrm{t}} = \sum_{i=1}^{N} (\Delta t_{\mathrm{heat,i}} + \Delta t_{\mathrm{cool,i}}), \tag{3.1}$$

where T_{t} is the total treatment time, N is the number of sonications, and $\Delta t_{\mathrm{heat,i}}$ and $\Delta t_{\mathrm{cool,i}}$ are the heating and cooling period for sonication i, respectively.

The primary reason for prolonged treatment times is the cooling times between sonications. They are necessary to avoid overheating in the near field [14, 16]. Mougenot *et al.* showed that there exist a linear relationship between the maximum temperature increase in the near field and the deposited surface energy density, with the slope following the tissue ultrasound absorptivity factor, regardless of the sonication depth and cell size. Since there is very little constructive interference taking place between the ultrasound waves at near field, the heating at skin depth does not depend on the phase of the waves, but only on the local average deposited energy. [15] Although these findings provide a simple way of approximating the near field heating by a sonication, the temperature rise depends highly on local perfusion rates, which in general are not well known during the treatment [11].

Based on these findings it could be concluded that sonicating deeper into the tissue implies shorter cooling times, since the deposited energy is distributed across a larger skin area. In contrast, the deeper the target, the more power is needed, increasing the surface energy. As the absorption rates are tissue and patient specific as well as heterogeneous, it is not a trivial task to optimize the depth levels of the sonications.

Several parameters influence the outcome of the treatment: cell sizes, cell position, sonication power, the heating time for each cell, the cooling time between each sonication, and the order in which the cells are sonicated. It is generally recommended to use as large

cells as possible, as it reduces the total number of sonications and therefore improves the efficiency of the treatment [11, 25, 26].

The choice of heating time and power are tightly intervened, and the balancing the two is not trivial. Most publications tend to discuss the optimization of heating times for ablating small treatment cells and not larger volumes [11, 27, 28]. Similar limitations are present in publications considering the optimization of cooling times [11, 23–25]. The main conclusion in these texts is that the problem is very complex and that, while simplified models might give approximative results [11, 24], the need for knowledge about tissue parameters and the use of dynamic algorithms is obvious [25]. The optimal solution depends, among other things, on the target, the tissue type and structures, the blood perfusion rates, and the transducer and its power deposition pattern [24, 25]. Payne argues [24] that if the choice lies between using a conservative heating strategy (using the minimal amount of power that is able to induce the required dosage in the target but prolonging the sonication time) and an aggressive treatment strategy (using the maximum amount of power and short heating times to induce the dose as quickly as possible), the aggressive strategy is to be preferred. Due to the nonlinear nature of thermal dose accumulation, a point of diminishing returns exists, below which the dose is actually delivered during the cooling phase and not during the heating phase. Payne also pointed out that optimal cooling and heating times does exist, but that they depend on the effective perfusion and power density in the normal tissue and the tumour, and that to be able to optimize the durations on-line, a rapid method for measuring or approximating these parameters is needed. The reason is that the perfusion rates in the normal tissue and in the target zone change during the treatment, and the optimization needs to be repeated several times. This also implies that a fast approximative model of the temperature rise is needed. As Payne's results were acquired by simulations, using an approximative exponential model for temperature change, it is unclear whether near field heating was taken into account. Mougenot *et al.* [26] attempted to optimize the cooling times for a set of cells with a predetermined sonication order and power levels. The result showed that by using relatively large volumetric cells and keeping the cooling time low, it is possible to take advantage of the cumulative heating of the prior sonications to increase the efficiency of the succeeding sonications. Mougenot found that it was possible to increase the necrosed volume in animal trials by a factor of five with lower power levels and without inducing skin damage by choosing the cooling time appropriately.

Sonication order, the order in which the cells are sonicated, is another aspect of HIFU treatment optimization that is sparsely researched. Malinen *et al.* [29] proposed a method based on the minimum time formulation of the optimal control theory for trajectory path optimization, but the simulations were done only on very small targets (radius 1.9 cm). Moreover, the simulation did not take into account near-field heating and was computationally heavy. As the results depend on heterogeneous tissue parameters, the optimization would require detailed information about the targeted tissue and would be too slow for clinical use. A general guideline given in [7] (and Philips MR-HIFU training material [30]) is to choose the order that minimizes the overlap and maximizes the distance between successive sonications in order to minimize the heat build up in the near field.

As with the optimization of a HIFU treatment, treatment planning as a whole is dealt with in only a few publications. White *et al.* [31] recognised the necessity of predicting the US propagation parameters in order to plan an effective HIFU treatment. In order

to achieve this, White proposed the use of either MRI or CT scans. Fedewa *et al.* [6] presented an automated treatment planning procedure for prostate cancer treatment. The first step of the procedure consists of acquiring images of the target and creating models of the structure. These models are then used to fill the target with planned cells according to some pre-set rules, starting with the largest possible cells and then moving down the scale. Apparently the algorithm does not perform any position optimization *per se*.

To conclude, it seems that the measurement of local ultrasound propagation parameters in and around the target volume is essential to be able to optimize a HIFU treatment. An attempt to achieve this using MR technology has been conducted by Dragonu *et al.* [10]. Their approach was based on quantitatively analysing the temperature maps acquired by the MRI scanner during the heating and cooling stage of a sonication. The technique allowed them to find good approximations of the absorption, perfusion, and diffusion rates for the target tissue. The model makes the simplifying assumption that the parameters are homogeneous, and thus only finds the average rates in the investigated area. It was recognised that the method could be applied to monitor changes in tissue parameters during the treatment, as a complement to the post-treatment verification scans.

4 Similar Planning Applications

As the number of published papers regarding the optimization of HIFU treatments is limited, this section introduces two optimization problems with similarities to that of MR-HIFU treatment planning. The emphasis is on the optimization techniques used.

4.1 Brachytheraphy

Radiotherapy is the use of ionizing radiation (such as photons and electrons) to treat cancer tumours. The radiation damages and destroys the cells in tissues subjected to it, be they malignant tumour cells or normal healthy cells. The aim of radiotherapy treatment planning is, therefore, to deliver a prescribed dose of radiation to the tumour, or the *primary/planned target volume* (PTV), while keeping the dose to the *normal tissue* (NT) and especially sensitive organs, OARs, to a minimum [32, 33].

Broadly speaking, radiotherapy can be divided into two different techniques based on how the radiation is delivered to the tumour: *External beam therapy*, also known as *teletherapy*, and *brachytherapy* [32]. In brachytherapy the radiation is applied by inserting radioactive sources or *seeds* into or close to the tumour with the help of catheters. This way, the radiation can be contained more easily, sparing more of the NT, than in teletherapy in which the radiation is irradiated from outside the patient. Brachytherapy can be further divided into two subgroups: high and low dose rate brachytherapy. In *low dose rate* (LDR) brachytherapy, radioactive seeds with low activity are inserted into the tumour. The activity of the seeds decreases quickly and the treatment is complete when all the seeds' activity has diminished. In *high dose rate* (HDR) brachytherapy, on the other hand, sources with a high activity are inserted only temporarily for a pre-set time (called the *dwell time*) [32, 33]. This approach allows the operator to optimize the treatment not only by altering the number and positions of seeds inserted into the patient, as is the case with LDR, but also by altering the dwell time of each seed [34]. Brachytherapy is used in the treatment of e.g. prostate, breast, tongue, and gynaecological cancers [33].

Due to the nature of the radioactive sources used in brachytherapy, the dose at a position $\bar{x}_{\mathrm{p}}$ from a seed located at position $\bar{x}_{\mathrm{s}}$, when $\bar{x}_{\mathrm{p}} \neq \bar{x}_{\mathrm{s}}$, depends approximately only on the radial distance from $\bar{x}_{\mathrm{p}}$ to $\bar{x}_{\mathrm{s}}$ and the type and the strength of the source. In HIFU, on the other hand, the spread of heat, temperature and thermal dose is more complicated, and depends on several tissue parameters such as perfusion rates and specific heat capacity. In brachytherapy, the cumulative radiation dose over time is the sum of the dose from each seed for every time instance that the respective seed is in place, taking into account the half-life of the seed [33]. Again, temperature does not accumulate as linearly over time due to perfusion. All in all, while HDR bears several similarities with HIFU therapy, the physical and physiological mechanism are quite different, implying that the mathematical models and simplifications used in HDR treatment optimization cannot be directly used in HIFU treatment optimization. The general treatment planning workflow in HDR is, however, well established and may well be used in MR-HIFU.

4.1.1 Treatment Planning in HDR

The treatment planning procedure in HDR starts with locating the tumour, by the use of MRI, CT or ultrasonography. Ultrasonography is also, in general, used to guide the placement of the implantation catheters. Furthermore, the high resolution imaging modalities can be used to extract detailed information about the anatomy in the *region of interest* (ROI), such as the form and position of the tumour and OARs. [33, 35] Once the anatomy of the ROI has been imaged the structures can be digitized and discretized by *segmenting* the acquired images. The discretized structures can be used in the construction of a mathematical model, which then is used to optimize the HDR treatment. The optimization can be carried out either by solving a set of equations which govern the dose at certain abstract *calculation points* or by using so called *expert systems* or *artificial neural networks* (ANN) .

In short, expert systems and ANNs are computational systems that can be trained to help in the decision making process in complex problems, making use of a vast database of knowledge and by using *artificial intelligence*. They are also able to learn based on the quality of their output. Expert systems and ANNs have in recent years gathered immense interest for the use as so called *clinical decision-support systems*, which aid the physicians to deliver a correct diagnosis based on the patient's data. They have also been implemented as treatment planning algorithms in brachytherapy, and in radiotherapy in general [36–39]. One of the main advantages of using expert systems over mathematical optimization techniques, especially in radiotherapy treatment planning, is the decrease in the computation time required to produce the plans. Solving the mathematical equations can take up to 1.5 hours, whereas a well trained expert system is able to produce good plans in seconds. [37] Even though the training of the system is time consuming, it is only required to be carried out once. A disadvantage of implementing expert systems is that the training requires a large amount of good plans done by experts on a wide range of anatomical cases, which are not available for new modalities such as MR-HIFU.

The mathematical optimization of the treatment is done by controlling the dose at a large set of calculation points which are spread out in the PTV, the OAR, and the normal tissue. Preferably the dose calculations could be done for each and every point in the ROI, but considering the required computation time this is in practice impossible. [32] The last step in the planning procedure, before the actual optimization is performed, involves prescribing the doses to each of the regions. The dose levels needed to cause tissue death inside the tumour are considered, as well as the safe levels that OARs can be exposed to [32, 33]. The result is a *mixed integer programming* model, with the objective to e.g. minimize the dose outside the PTV, and with constraints controlling the dose inside the PTV. Binary variables are used to indicate which seed positions are chosen; positive real variables represent the dwell times. Due to the inherent differences in the HDR and MR-HIFU dose delivery, a detailed description of the mathematical models used for optimizing the HDR treatments will not be given here.

Several different optimization algorithms have been used to optimize the MIP problems in HDR treatment planning. Deterministic branch-and-bound (BB) techniques [33] and similar branch-and-cut (BC) algorithms can guarantee to find an optimal solution to the problem, but require in general longer computation time to find the solution. Probabilistic techniques, such as simulated annealing (SA), genetic algorithms (GA) and tabu

search, explore the solution space in a more random matter while trying to guide the search towards the global optimum [32–34]. As these methods involve probabilistic decisions regarding in which direction to move the search, and as the solution space often is non-linear and contains several local minima, these methods cannot guarantee that the best solution is found. A more thorough description of BB, GA and SA can be found in Section 6. More elaborate multi-objective optimization methods have also been implemented [40], which is useful in brachytherapy-like applications where many, often conflicting objectives can be identified.

4.2 Network Planning

As the treatment doses in MR-HIFU are delivered in the form of well-confined cells, the problem structure resembles optimization of wireless networks. The aim of network planning is to optimize networks in such a way that their coverage is maximized and cost minimized. Networks are defined here in a very broad sense, with most cases originating in wireless communication networks [41–51], network traffic monitoring [52, 53] and networks of surveillance sensors [54, 55].

Network optimization has spurred much interest in the academic world. A crucial part of the network optimization process is finding optimal locations for the base stations, network monitors, sensors *etc.* from a list of possible candidate locations. This is a classical combinatorial optimization problem and is known to be *NP-hard* [53]. A more detailed discussion about *NP*-hard problems can be found in Section 5.1, but in short *NP*-hard problems are problems that cannot be solved exactly in reasonable time, and their solution can only be approximated. The network optimization problems presented in literature can be very complicated, taking into consideration capacity, demand, sampling rates *etc.*, but in general two different sub-problems can be identified: *budgeted coverage maximum* (BMC) [52, 53, 55] and *minimum cost with coverage threshold* (MCCT) [41, 53, 54, 56]. The former aims to maximize the coverage of the network in such a way that the total installation cost of the base stations does not exceed a maximum budget limit. Variants of this include maximizing the percentage of supplied customers that is covered [42] or maximizing the number of customers that are served [49], which in essence adds a weight distribution to the problem. The latter problem is the dual of the cover maximization problem. The aim is to minimize the cost of the installed base stations in a network which supplies either all of the customers in the area [54] or a given percentage of the customer base [47, 49]. Often the decision between low cost and high coverage is non-trivial, and multi-objective optimization models have, therefore, been developed [43–46, 50].

As both the BMC and the MCCT problems can be expressed as *integer programming* (IP) problems (see Section 5.1) most of the algorithms used in HDR treatment planning have also been implemented in network planning: heuristic greedy algorithms [41, 49, 53], metaheuristic simulated annealing [46] and tabu search [47], and deterministic branch-and-cut algorithms [56]. The most popular family of algorithms are the genetic algorithms, also a randomized metaheuristic [42–45, 48, 50–52, 55]. These algorithms, excluding tabu search, are described in more detail in Section 6

5 Combinatorial Optimization Problems

Selecting the optimal subset of cells for a given MR-HIFU treatment case is a combinatorial optimization problem, resembling the problems encountered in network planning, as described in Section 4.2. It also turns out that selecting the best sonication order is also a combinatorial problem. Many combinatorial optimization problems are very difficult to solve. Often, deterministic algorithms, guaranteed to find the global optimum for the problem, are computationally very heavy, as the number of different combinations grows rapidly with the number of elements in the problem.

In order to classify the complexity of problems, they are often divided into two classes: *P* and *NP*. Class *P* (*deterministic polynomial time*) problems are problems that, using an appropriate algorithm, can be solved in *polynomial* or *reasonable time*. Class *NP* (*non-deterministic polynomial time*) problems, on the other hand, do not necessarily need to be solvable in polynomial time but, if the solution of the problem is known, the correctness of the solution can be verified in polynomial time. *NP* problems are decision problems, i.e. the output is either YES or NO. All *P* problems are also *NP* problems but it is widely assumed – although not proven – that $P \neq NP$. In practice, if the solution for a *NP*-problem is not known, an exhaustive search of the solution space is needed in order to find it. [57]

Class *NP*-hard problems are at least as hard as *NP* problems. Proving that a problem is *NP*-hard is usually done by proving that it is harder than another problem that is already known to be *NP*-hard. A *NP*-hard problem is not required to be *NP* decision problem, which is the case for many optimization problems that return a set of numbers. [57]

To summarize, many combinatorial optimization problems, and all of the problems presented the sections below [53, 58–60], are *NP*-hard and therefore so complex that no algorithms exist that solve large instances of them in polynomial time. Exhaustive search of the solution space would, in theory, find the solution, but evaluating every position in the solution space in order to find the optimum is not feasible. For this reason, approximative or probabilistic algorithms, which find feasible and hopefully good solutions but not necessarily the optimum, are used. Examples of such algorithms are presented in Section 6. This section presents two combinatorial problems relevant for MR-HIFU treatment planning: *covering problems* and *routing problems*.

5.1 Covering Problems

Covering problems are here defined as follows: design optimally a combinatorial structure so that it covers another, subjected to a set of constraints. The budgeted maximum coverage (BMC) and the *budgeted unique coverage* (BUC) problems aim to maximize the coverage of a set of *elements* by choosing an optimal combination of *sets* or *cells*, each with an associated cost, from a family of sets, restricted by a budget. The duals of the problems, the minimum cost with coverage threshold (MCCT) and the *minimum cost with unique coverage threshold* (MCUCT) problems, aim to minimize the cost of the chosen sets, provided the coverage provided by the sets exceeds a given threshold. In MR-HIFU population optimization, the sets are cells and the elements are discrete points representing the target volume. Algorithms used to solve combinatorial algorithms are presented in Section 6.

5.1.1 Budgeted Maximum Coverage

Khuller *et al.* [59] defines the BMC problem as follows: given a family of sets $\mathcal{S} = \{S_1, S_2, ..., S_m\}$, with associated costs $\{c_i\}_{i=1}^{M}$, which is defined over a domain of points $\mathcal{X} = \{x_1, x_2, ..., x_n\}$ with associated weights $\{w_j\}_{j=1}^{N}$, find the optimal subfamily $\mathcal{S}' \subseteq \mathcal{S}$ of sets so that the total weight of the covered points is maximized and so that the total cost of the elements in $\mathcal{S}'$ does not exceed the budget B.

The problem can be expressed as an integer programming model as follows

$$max \sum_{j=1}^{N} w_j x_j, \tag{5.1a}$$

$$\text{subject to} \sum_{i=1}^{M} c_i y_i \leq B, \tag{5.1b}$$

$$x_i, y_i \in \{0, 1\}, \tag{5.1c}$$

where w_j is the weight of point j, x_j is a boolean indicating if point j is covered or not, N is the number of points, c_i is the cost of cell i, y_i a boolean indicating if cell i is chosen or included in $\mathcal{S}'$, M is the number of cells and B is the budget. The objective function (5.1a) denotes the maximization of the weighted sum of the covered points. The constraint (5.1b) dictates that the total cost of the chosen cells cannot exceed the budget. It should be noted that, even though each point j can be covered by any number of cells, covering the same point with several sets does not increase the value x_j in the objective function. The BMC problem has been applied in several fields, for example in facility and base station location problems [42, 53, 61], in network optimization and query planning [62, 63], and in problems related to sensor and monitoring [52, 55] optimization.

5.1.2 Budgeted Unique Coverage Problem

Demaine *et al.* [64] defines the BUC problem as follows: given a family of sets $\mathcal{S} = \{S_1, S_2, ..., S_m\}$, with associated costs $\{c_i\}_{i=1}^{M}$, which is defined over a domain of points $\mathcal{X} = \{x_1, x_2, ..., x_n\}$ with associated weights $\{w_j\}_{j=1}^{N}$, find an optimal subset $\mathcal{S}' \subseteq \mathcal{S}$ so that the total weight of the covered points that are *uniquely covered* is maximized and so

that the total cost of the elements in $\mathcal{S}'$ does not exceed the budget B.

$$max \sum_{j=1}^{N} w_j x_j, \tag{5.2a}$$

$$\text{subject to} \sum_{i=1}^{M} c_i y_i \leq B, \tag{5.2b}$$

$$\sum_{i=1}^{M} x_{ij} \leq 1 \quad \forall j, \tag{5.2c}$$

$$x_i, x_{ij}, y_i \in \{0, 1\}, \tag{5.2d}$$

where x_{ij} is a boolean indicating whether point j is covered by cell i. The objective function (5.2a) denotes the maximization of the weighted sum of the covered points. The constraint (5.2b) dictates that the total cost of the chosen cells cannot exceed the budget, while the unique coverage constraint (5.2c) indicates that each position can be covered by at most one cell. An example of the problem is visualized in Figure 4. The unique coverage problem has been applied in network planning and in so called unlimited-supply single-minded pricing, a problem of optimizing item prices to maximize the seller's profit [64].

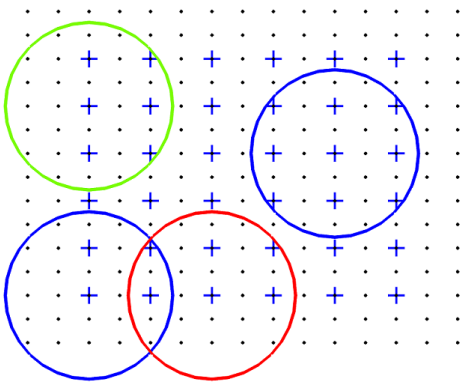


Figure 4: An example of the BUC problem. The black dots signify points that are to be covered, while the blue circles are two chosen cells. The red cell cannot be chosen, as it overlaps with one of the blue cells. The green cell, on the other hand, does not overlap and can therefore be chosen.

5.1.3 Minimum Cost with Coverage Threshold

The MCCT problem is defined as follows: given a family of sets $\mathcal{S} = \{S_1, S_2, ..., S_m\}$, with associated costs $\{c_i\}_{i=1}^{M}$, which is defined over a domain of points $X = \{x_1, x_2, ..., x_n\}$ with associated weights $\{w_j\}_{j=1}^{N}$, find an optimal subset $\mathcal{S}' \subseteq \mathcal{S}$ so that the total cost of the chosen cells is minimized and the total weight of covered points exceeds a threshold T.

The MCCT problem can be described as an IP problem as follows [63]

$$min \sum_{i=1}^{M} c_i y_i, \tag{5.3a}$$

$$\text{subject to} \sum_{j=1}^{N} w_j x_j \geq T, \tag{5.3b}$$

$$x_i, y_i \in \{0, 1\}, \tag{5.3c}$$

where T is the threshold for the weighted coverage. The objective function (5.3a) denotes the minimization of the total cost of the chosen cells. The constraint (5.3b) dictates that total weighted coverage should exceed the threshold T.

Adding the further constraint of unique coverage of the elements (Equation (5.2c)) to the MCCT problem gives the MCUCT problem. To the writer's knowledge, the MCUCT problem has not been presented before in literature. A detailed investigation of the approximability of the problem is beyond the scope of this thesis, but as the MCUCT problem is further constrained than the MCCT problem, it is also at least *NP*-hard. The field of application for the MCCT problem are much the same as for BMC, albeit its applications have not been dealt with as extensively in literature [53, 63].

5.2 Routing Problems

The optimization of the sonication order is a typical routing or sequencing problem. Consider a graph, consisting of vertices or nodes *V*, connected by arcs *A*. The arcs have associated costs or distances c. *Routing problems*, in general, consists of optimizing the circuit in the graph. In this section, we first consider the best known problem of the type, the *travelling salesman problem* (TSP), which has been dealt with extensively in literature and for which most algorithms have been developed. After that, a problem more relevant to MR-HIFU treatment planning is presented.

In the TSP, the task is to minimize the total length of the route that passes through each node once and only once, starting and ending at the same node. Such a route is also called a *tour* or a *Hamiltonian cycle*. In the TSP graph all nodes are, thus, connected with each other by arcs A_{ij}, where the ij subscript signifies that the arc starts at node i and ends at node j, and each arc has a cost or a distance c_{ij}. If the cost of the arc connecting V_i and V_j depend on the direction one travels along the arc ($c_{ij} \neq c_{ji}$) the TSP is said to be *asymmetrical* (ATSP). If $c_{ij} = c_{ji}$ for all arcs in the graph, the problem is said to be *symmetrical* (STSP). Furthermore, if the distances satisfy the triangle equality ($c_{ij} + c_{jk} \geq c_{ik}$) the problem is said to be Euclidean. [65]

The TSP can be formulated as an integer programming model as follows [65]

$$min \sum_{i \neq j} c_{ij} x_{\mathrm{a},ij}, \tag{5.4a}$$

$$\text{subject to} \sum_{j=1}^{N} x_{\mathrm{a},ij} = 1 \qquad i = 1, \ldots, N, \tag{5.4b}$$

$$\sum_{i=1}^{N} x_{\mathrm{a},ij} = 1 \qquad j = 1, \ldots, N, \tag{5.4c}$$

$$x_{\mathrm{a},ij} \in \{0, 1\}, \tag{5.4d}$$

where $x_{\mathrm{a},ij}$ is a boolean determining whether arc A_{ij} is chosen or not, and N is the number of nodes. Equations (5.4b) and (5.4c) specify that every node is entered and left exactly once, respectively. In addition to model (5.4), so called *subtour elimination constraints* are required. These eliminate tours that have a length shorter than N and thus guarantees that a single tour is produced. Subtour elimination constraints are required if the TSP is to be solved using normal IP solving techniques, but as they tend to increase the complexity and size of the problem considerably, and as several other solving techniques do not require them [65], they will not be described here in any further detail. Applications for the TSP are numerous, for example routing problems and job sequencing [65].

A problem that resembles the TSP is the *longest (Hamiltonian) path problem* (LPP). In the LPP, the aim is to maximize the length of the path which passes through every node exactly once. LPP differs from TSP in that it is a maximization problem and it does not seek a complete tour but instead a path that ends at the last node, without returning to the first. Even though LPP is not as thoroughly studied in literature and Wu *et al.* [66] proclaims that it is doubtful that the problem has an approximation algorithm with a constant performance ratio, algorithms developed for TSP problems can often be easily altered to solve the LPP. Algorithms for solving the TSP and LPP are presented in Section 6.

Similarly to TSP, an IP model of LPP can be described as follows

$$max \sum_{i \neq j} c_{ij} x_{\mathrm{a},ij}, \tag{5.5a}$$

$$\text{subject to} \sum_{j=1} x_{\mathrm{a},ij} = 1 \qquad i = 1, \ldots, N, , i \neq k, \tag{5.5b}$$

$$\sum_{j=k} x_{\mathrm{a},ij} = 0 \qquad i = k, \tag{5.5c}$$

$$\sum_{i=1} x_{\mathrm{a},ij} \leq 1 \qquad j = 1, \ldots, N, , j \neq k, \tag{5.5d}$$

$$\sum_{i=1} x_{\mathrm{a},ij} = 1 \qquad j = k, \tag{5.5e}$$

$$x_{\mathrm{a},ij} \in \{0, 1\}, \tag{5.5f}$$

where k is the index of the first node in the path. As with the TSP, if this model is to be solved using regular IP solving methods extra subtour elimination constraints need to be implemented. Furthermore, the model needs to be solved N times, each time with a different k value, as the maximum path length depends on where the path starts. Equations (5.5b) and (5.5c) dictate that all nodes are entered exactly one time while the first node is never entered. Moreover, Equations (5.5d) and (5.5e) specifies that the first node is left once, while the other nodes are left no more than one time. This is due to the fact that the last node, which is never left, is not known. LPP has applications in e.g. data clustering [67].

6 Algorithms for Combinatorial Optimization Problems

As the combinatorial problems presented in Section 5 are *NP*-hard, approximative algorithm are required to solve large instances of them. Several families of algorithms for solving combinatorial optimization problems exist. Four very popular ones are presented in more detail below. The first family, greedy algorithms, are simple and fast algorithms that can perform surprisingly well, if the problem structure is suitable. *Greedy randomized adaptive search algorithms* (GRASP) is a family of algorithms which tries to combine the fast solution construction abilities of greedy algorithms with the proven capabilities of the deterministic local search algorithm. The third family, genetic algorithms (GA), are a group of algorithms that imitate natural evolution through the survival of the fittest. The key idea is to work on not only one but on a set of solutions in a guided randomized search. Branch-and-bound (BB) algorithms, in contrast, are deterministic and are guaranteed find the global optimum of the problem. For large problems the disadvantage is the impractical computation time.

Simulated annealing is another algorithm which imitates a physical phenomenon, the annealing of metals. It resembles the genetic algorithms, but only works on one solution at a time. The algorithm introduces random changes to the current solution, evaluates the solutions goodness and then decides, based on a probability that is a function of the algorithm progression, if the new solution is kept or discarded. On large and complex problems, simulated annealing has been shown to not only perform almost as good as or even better than genetic algorithms result wise but also to find good solutions faster [68].

This section presents the aforementioned algorithms by starting with a general introduction of each algorithm, followed by description of how they are implemented in covering and routing problems, respectively.

6.1 Greedy Algorithms and GRASP

Greedy algorithms are a large and popular family of heuristics for combinatorial problems, even though they seldom offer a performance guarantee. Their popularity lies in their simplicity: they are easy to implement and they are computationally fast. A greedy algorithm constructs a feasible solution to a problem by iterating through a set of steps, and making a choice at each step about how to continue the construction process. The choice made is always the one that seems best at the time, i.e. the algorithm strives towards the local optimum at each step. After each step and before the algorithm makes the next choice, the set of allowed choices is updated. Even though greedy algorithms may seem quite naive, and fail to find optimal solutions for many problems, they are capable of finding *good enough* solutions for many applications. [69]

A GRASP algorithm is a repetitive process, which creates a number of feasible solutions and selects the best one to be the final solution. Each iteration consists of two phases: a *construction phase* and a *local search phase*. GRASP bears a close relation with greedy algorithms, as the solution candidates are built in much the same way in the construction phase, greedily adding new elements to the solution one at a time and adaptively updating the goodness of the remaining elements after each iteration. The construction phase of the GRASP algorithm differs from the greedy algorithm in that it does not necessarily

select the best choice, but it randomly selects *one of the best* choices. As a consequence, the construction phase produces different solutions in each iteration. Next the solution is improved through a local search algorithm, which searches for the local optimum in the neighbourhood of the current solution. All of the candidate solutions produced by each iteration are compared and the best one is returned. [70] As the construction phase is as fast as a greedy algorithm, the bottleneck of the GRASP algorithms computational effectiveness is the local search algorithm. GRASP algorithms have been applied in a range of combinatorial problems, such as network optimization problems [70, 71] and other location problems as well as scheduling and routing problems [71], but will in this thesis only be applied for coverage problems.

6.1.1 Greedy and GRASP Algorithms for Covering Problems

There are many publications on the use of greedy algorithms in network optimization. [41, 53, 61, 64]. A greedy algorithm designed for the BMC problem begins with ordering the available cells according to weighted coverage-to-cost ratio and selects the cell with the highest ratio. To take into account the area covered by the already chosen cell, the weighted coverage-to-cost ratios of the remaining cells are updated. After that, the cell with the highest ratio is again chosen. This procedure is repeated until there are no more cells to select, or as long as the budget is not exceeded. [63] Curtis *et al.* [62] proposed to run two greedy algorithms, one that aims to maximize the weighted coverage-to-cost ratio and another that strives to maximize only the weighted coverage, to solve a budgeted coverage maximization problem which penalized overlapping. The better of the two candidate solutions is chosen to be the final solution. The algorithm designed by Khuller *et al.* [59], which demonstrably gives the best possible approximation ratio for the BMC problem (unless *P=NP*), was based on the same ratio heuristics, with minor additions to deal with special cases.

A greedy algorithm for the MCCT problem resembles the algorithm for the BMC problem, the only difference being the ordering criteria. For the MCCT problem, the cells are ordered in ascending order according to their cost-to-weighted coverage ratio, and the algorithm selects the cells with the smallest ratios first. The algorithm selects new cells as long as the threshold constraint is not fulfilled.

While greedy algorithms, such as those presented above, have been proven to perform well for BMC and MCCT problems as constructors of good, albeit not optimal solutions, their performance depends on the structure of problem. Notably, both the MCCT and BMC heuristics seem to perform better on problems in which the costs and weights are heterogeneously distributed. This can be intuitively understood, as it makes the choice of cells with most coverage easier. [63]

6.1.2 Greedy Algorithms for Routing Problems

Greedy-like algorithms in the TSP are so called *tour constructing heuristics*, which simply constructs a tour and then stops, without attempting to improve on the tour afterwards. Two of the most popular greedy-like algorithms used for tour construction in TSP are *nearest neighbour* (NN) and *greedy* (GR) [72]. NN starts at a random node and chooses

the nearest neighbour that has not yet been visited to be the next node. This procedure is continued until a complete tour has been constructed. GR starts by ordering all arcs in ascending order according to their length or cost. Next, it starts selecting the shortest arcs, skipping the ones that would cause subtours, until the tour is complete. [72–74] Despite their simplicity, the heuristics have shown good results as approximative algorithms for Euclidean TSP [75, 76]. In empirical tests performed on large Euclidean TSP (10^4 nodes) GR have shown only slightly better performance and running time than NN [73].

The basic principle of the TSP greedy algorithms can intuitively be applied to the LPP, even though scientific reports on the subject are sparse. A GR heuristics for the longest path problem begins constructing the path from the other end of the order list of arcs, stopping when the path is complete. The analogue of the NN is a *farthest neighbour* (FN) heuristics, which starts similarly to the NN at a random node, but construct the path by continuing to the neighbour farthest away from the current node.

6.2 Genetic Algorithms

Genetic algorithms are a loosely defined family of randomized guided algorithms, which replicates the behaviour of genetics and natural selection in the biological world [77, 78]. GAs have been shown to work well on complex optimization problems where deterministic methods fail. For other sorts of problems that are well-defined and have a smooth solution space, deterministic methods still work more efficiently. [78] Genetic algorithms have had vast areas of applications ranging from solving complex mathematical problems, to optimization in technological areas such as machine learning and to be used as simplified scientific models in economics, ecology and sociology [77]. More complex implementations of GAs have introduced multi-objective optimization, or parallelization [48], or using heuristics to fine tune chromosomes and/or feed the algorithm with non-random initial population. [78]

Genetic algorithms were first introduced by Holland in 1975 [78]. Even though the concept of GAs is loosely defined and a vast number of variants exist, they all have a few important common features. As mentioned, genetic algorithms does not work on only one feasible solution per iteration, but on a set of solutions, called a *population*[1] or a *generation*. [77] The algorithm aims to evolve this set, according to a set of rules, towards a maximum *fitness* [77, 78]. A feasible solution is called a *chromosome* or an *individual*, with the nature of the solution encoded in it. The chromosomes are tested for their *fitness*, by inserting them into a *fitness function*. Every chromosome represents a point in the solution space, and its fitness is the value of the solution space at that point. The chromosomes with high fitnesses are used as a parent for the next generation. The mating process is carried out by *crossover*, taking one part of one parent and another part of the other parent and combining them to create a child. If the two parents have good genes chances are that their child, being the combination of the two, will have an even better fitness. Lastly, as is the case with biological genes, *mutation* may occur in a chromosome with a certain probability. This mutation randomly changes a number of genes in a chromosome, bringing new diversity into the population that might otherwise be too similar to their ancestors and hindering

[1] The term *population* here should not be mistaken for the *population* process, i.e. the filling of treatment cells inside the target volume in the context of HIFU treatment planning.

the development. This is the analogy of getting caught in a local minimum. [77] As the GAs has a good ability to escape local minima and as they sample the solution space at several points simultaneously, they are well suited for problems with very complex solution spaces. Other advantages include no need for derivative information, being well suited for parallelization, and providing a set of good solutions and not only one. [78]

The process inside the GA is essentially quite simple but very powerful. The most challenging aspect of solving problems using GAs is often finding a suitable way of encoding the chromosomes and designing the crossover, mutation and other operators so that each point in the solution space can be represented and reached. Traditionally, chromosomes have been encoded as bit-strings that are used either to represent decision variables (boolean, yes or no) or numbers. Encoding the chromosome as continuous values is also possible. [77, 78]

In its most simple form, a GA is initiated by creating a first population of chromosomes at random. Next, the fitness of the chromosomes is evaluated by the fitness function, which is the function being maximized by the algorithm. After that, the best chromosomes are chosen to be parents for the next generation. [77, 78] This can be carried out in numerous different ways, but a commonly used selection type is called *roulette wheel selection* [51, 77]. In roulette wheel selection, each chromosome is given a portion on the roulette wheel, or probability of being selected, proportional to its fitness. If the roulette ball stops at a given portion, the respective chromosome is selected. This means that more fit chromosomes have a better chance of becoming chosen, than the less fit. The reason for not simply choosing the best chromosomes directly is to preserve the diversity in the population and prevent the algorithm from converging to a local minimum prematurely. When the parents have been chosen, the crossover operator is initialized. In the simplest type of crossover, *single-point crossover* [51, 77, 78], a crossover point is chosen by selecting an interval between two bits in a string at random. A child is created by combining the parts of the first and the second parent that precedes and succeeds the crossover point, respectively. This procedure is repeated until a new population has been created. Finally, each bit in each chromosome is flipped, or mutated, with some small probability. The *mutation probability* is usually quite small, small enough not to destroy the good genes of the fit parents in the next generation but large enough to introduce diversity into the population, to be able to escape local minima. [77] It should be noted that the success and effectiveness of a GA is highly dependent on how the operators are designed and on the parameters that control the operators (such as the population size and mutation probability). Unfortunately optimization of these parameters has proven quite difficult. [44, 77]

Even though genetic algorithms have been applied in a wide range of problems, it is still unclear what the basis of the effectiveness is. According to Holland, the power of GAs lies in their way of working with *building blocks*, or *schemas*, for a good solution. In the case of bit-strings, schemas consist of ones and zeros at some of but not all positions in the string. The positions that are not occupied by ones of zeros, can take on either value. In one sense, schemas are hyperplanes in the solution space. The GA evaluates and compares an enormous amount of building blocks for a good solution in parallel, and the number of occurrences of good schemas will statistically increase as the algorithm propagates due to the bias of the selection operator towards higher than average fitnesses. [77]

6.2.1 Genetic Algorithms for Covering Problems

More complex covering problems, such as network optimizing problems, have been solved with sophisticated genetic algorithms, utilizing, e.g. variable length chromosomes [42] or multilevel encoding [45, 50], using complicated fitness functions taking into account capacities and demands [50] or radio wave propagation models [44, 50], and local search heuristics to improve the performance [42]. In this thesis a traditional fixed-length binary string representation is used to encode which cell is chosen from the set S into the subset S'. Below are presented some of the alternative operators to be used in a GA, designed to work for BMC, BUC, MCCT and MCUCT problems using binary encoded chromosomes.

Initialization The initialization of the first population can either be done randomly, which is the traditional method, or heuristically, which is *seeding* the algorithm with a population that is known to have better average fitness than a random population. Non-random initialization can be produced e.g. by issuing a local search operator on the randomly initialized population or utilizing a greedy-like algorithm to find neighbourhoods for good solutions. [55] This so called seeding method can take advantage of *a priori* knowledge about the problem to improve the start of the algorithm. The disadvantage of using seeding is that it can, if not used carefully, lead to premature convergence. [79]

Selection After the fitness of the chromosomes has been determined, the selection procedure selects the chromosomes to be used as parents for the next generation. The roulette wheel selection, explained in Section 6.2, is quite simple to implement. The difficulty lies in determining how the probabilities should be distributed among the candidates. If the probabilities are simply assigned according to the fitness of the chromosomes, and the good chromosomes have superior fit compared to the rest of the population, the superior ones will reproduce very quickly, possibly guiding the algorithm towards a local optimum. For this reason, the good chromosomes reproduce quickly in the beginning when the fitness variance in the population is large, but poorly when the algorithm converges towards an optima and the fitness variance in the population is smaller. The *selection pressure* is said to vary with the fitness variance. [77] One way of alleviating this problem is to use so called *sigma scaling*, as introduced by Mitchell [77]. Sigma scaling scales the fitness values of the population with the standard deviation and the average of the fitnesses, and thus maps the fitnesses to expectation values that are less sensitive to changes in the fitness variance. This keeps the selection pressure relatively constant throughout the whole search. The scaling is done as [77]

$$ExpVal(i,t) = \begin{cases} 1 + \frac{f(i,t) - \bar{f}(t)}{2\sigma(t)} & \text{if } \sigma(t) \neq 0, \\ 1 & \text{if } \sigma(t) = 0, \end{cases} \tag{6.1}$$

where $ExpVal(i,t)$ is the expectation value for chromosome i at generation t, $f(i,t)$ is the fitness value of chromosome i, $\bar{f}(t)$ is the mean fitness of the population, and $\sigma(t)$ is the standard deviation of the fitnesses in the population. If the standard deviation of the population is zero, no scaling is done, as all the chromosomes have the same fitness. [77]

Another problem associated with roulette wheel selection is that if the populations are very small, and the roulette wheel is spun only a few times, there is a chance that the good

chromosomes are not chosen at all. This problem is solved by using *Stochastic universal sampling* (SUS). In this version of roulette wheel selection, the wheel is spun only once but instead of one ball the wheel has n equally spaced needles, where n is the number of chromosomes to be selected. All chromosomes on which one of the needles point are selected. As a consequence, the distribution of the selected chromosomes correlates more closely with the probability distribution. [77]

To further enhance the performance of the algorithm and to ensure that the best chromosomes are not destroyed by the crossover and mutation operators, *elitism* is usually used. Elitism means that the algorithm selects the best chromosomes to be moved directly to the next generation. They have the possibility to act as parents, but they are also given a position in the next generation. The number of chromosomes that are set to be elite should be chosen with care, not to cause diversity to decrease in the population. [45, 77, 78]

Crossover As explained in Section 6.2, the most simple crossover operator is the single-point crossover operator. Although being simple and easy to implement, the method has bias to retain short schemas. As the algorithm chooses only one breakage point, schemas that have their ones and zeros on the only one side of the breakage point are retained, while schemas with elements at the far ends of the string are almost always discarded. [44, 77, 78]

Uniform crossover was designed to overcome this lack. In uniform crossover each bit in a given parent stands a given *crossover probability* to be used in the crossover. A given number of bits are chosen from the first parent to partake in the crossover, and are copied to the child. The remaining positions in the child's chromosome, which have not been filled by bits from the first parent, are filled by copying the corresponding bits in the second parent. This means that the method has no positional bias while still being simple to implement. While the choice of the best crossover operator is a complicated matter, uniform crossover is among the most popular and most extensively used. [44, 77].

Mutation While crossover is the main source of variation in a genetic algorithm, mutation is an important operator to reintroduce diversity into the population. Mutation is usually performed by flipping each bit with a set *mutation probability*. The mutation probability is kept low, around 1-2 %, not to destroy the newly created offspring. [77, 78]

Fitness functions and constraints The fitness function is central to the performance of the algorithm. For BMC the fitness is the sum of the weighted volume/area covered by each cell, as

$$f = \sum_{i=1}^{M} \sum_{j=1}^{N} y_i \, x_{ij} \, w_j, \tag{6.2}$$

where f is the fitness value, y_i is the binary value of the cell i having been selected or not, x_{ij} binary values communicate whether a voxel/pixel i is covered by cell j, w_j is the weight of the voxel/pixel j, and M and N are the number cells and voxels/pixels, respectively. As genetic algorithms are designed to maximize fitness values, the respective fitness value for MCCT is the reciprocal of the total cost of a set of chosen cells, and is given by

$$f = \frac{1}{\sum_{i=1}^{M} y_i \; c_i}, \tag{6.3}$$

where c_i is the cost of cell i.

The overlap constraint as well as the budget and threshold constraints set on the BUC and MCUCT problems respectively require constraint handling. As the normal crossover and mutation operators do not take into consideration these constraints, the operators which can produce infeasible chromosomes are therefore said to be *open*. In contrast, *closed* operators are guaranteed to produce only feasible chromosomes. If it is not possible to design closed operators, chromosomes not fulfilling all constraints should either be repaired, using a repair operator, or penalized by reducing their fitness value [42, 43, 45, 51]. Repair operators are problem-specific, and are used when a new population has been created to ensure that all chromosomes are feasible. Repair operators that are computationally efficient and robust for large problems can however be difficult to design [51]. Penalties can either be equal for all violators or depend on the extent of the violation. [80] This method is computationally less heavy, but may require more generations and larger population sizes to converge. The best method is problem specific.

6.2.2 Genetic Algorithms for Routing Problems

Due to TSP's popularity, several different encoding methods with accompanying operators have been developed for routing problems. Only the most intuitive and most widely used encoding method, *path representation*, is presented below.

The path representation encodes the path so that the index of each node is put in the position in the string in which order it is visited. Even though many closed operators for routing problems have been developed and while some operators have gained more popularity than others, their comparative performance has been scarcely studied. [79] The crossover operator used in this thesis, the greedy crossover, is explained in further detail in [81]. An detailed description of the other operators is beyond the scope of this thesis.

In order to improve the performance of GAs for routing problems, elitism and local search operators are in most cases implemented [79]. A popular local search (LS) method is the so called *2-opt local search*, which iterates through all subpaths in the path, reverses them and evaluates which order gives the best fitness value. More complex methods have shown to give better results, but are, on the other hand, computationally more demanding. [82] These improvement operators can be used at the beginning, at the end and/or during the iterations or every n-th iteration. Also the use of seeding is popular. Furthermore, an interesting stochastic operator, designed to introduce diversity into a population after it has reached a local optimum, is the *judgement day* (JD) operator, described by Kureichick *et al.* [83]. The JD operator re-initializes the population randomly, while keeping only the best chromosome in the population.

The encoding and operators used for genetic TSP algorithms can be utilized for LPP and other routing problems as well, but the fitness function needs to be altered to value longer paths higher than shorter ones. Furthermore, as the path in the LPP does not return to the first node after having visited the last, the fitnesses are calculated a slightly different manner.

6.3 Branch-and-bound Algorithms

Branch-and-bound algorithms are deterministic algorithms commonly used for solving integer programming problems [84]. They can also be applied to solving smaller TSP and other routing problems. Even though BB attempts to reduce the solution space and consequently reduce the computation time, the algorithm is exact and thus impractically slow to solve large *NP*-hard combinatorial problems. On the other hand, for small problems they can be quite fast, and they are guaranteed to find the global optimum.

The underlying concept of BB algorithms is based on dividing, or *branching*, the original IP problem into relaxed sub-problems which then are solved to obtain an upper bound (for maximization problems) on the objective value of the sub-problems. If the value of the solution to a sub-problem is less than the value of another feasible solution to some other sub-problem, then the optimal solution of the original IP problem cannot lie in the subset of the constraint set associated with the given sub-problem. [84] In practice this means that the sub-sub-problems, branched from the given sub-problem, are not required to be investigated, and an exhaustive search is not needed.

In the case of IP solving, a typical BB algorithm could propagate in the following manner. The initial IP problem is relaxed into a regular linear programming (LP) problem. As the LP problem does not require the variables to be integers, the solution to the problem might contain variables that are not integers. If all values are integers, the found solution is the optimal solution and the algorithm can stop. However, if one or more variables are not integers, the algorithm commences the first level of branching. Depending on the branching strategy, one of the non-integer variables is chosen to be the target of further constraints in the following sub-problems. Two different sub-problems are created, the first adding the constraint on the target variable to be less or equal to the variable rounded downwards, and the other adding the constraint on the target variable to be greater or equal to the variable rounded upwards. These problems are then solved again as regular LP problems, and the solutions are checked. If the solutions only contain integers, a feasible solution has been found, and the objective value of the solution is set as the current upper bound (the best known solution at this point). If, however, the solution contains non-integer variables, the problem is again branched and the new sub-problems are solved. Once a feasible solution is found, the algorithm returns one level up in the branching order and selects the next non-integer variable to be the next target for added constraints. If, during the branching, a solution is found which contains non-integer variables, but has an objective value less than the current upper bound, the branching is not continued, as it is certain that adding more constraints on the sub-problem will not lead to an increase in the objective value. Therefore, further branching is unnecessary. [84]

6.3.1 Branch-and-bound Algorithms for Coverage Problems

BB algorithms designed for IP problems can readily be used to solve the coverage problems presented by Equations 5.1, 5.2, and 5.3. By simply limiting the variable to values between 0 and 1, the IP problem solves the binary variables x. In practice, BB algorithms are usually extended with so called *cutting-plane* methods to form branch-and-cut (BC) algorithms. In short, the cutting plane method adds further inequality constraints on the relaxed LP model so that the infeasible fractional variable is *cut* out from the allowed solution space.

This reduces the solution space more effectively and thus also reduces the computational time required, compared to the regular BB algorithm. [56]

6.3.2 Branch-and-bound Algorithms for Routing Problems

BB algorithms have gained more popularity as solvers of routing problems, such as the TSP. Their popularity lies in that they are able to solve the routing problems without the need for formulating complicated subtour elimination constraints, as is the case if the problems are to be solved using common IP solvers. A popular BB algorithm for routing problems initially solves the IP model described by Equation (5.4) with the relaxation of not prohibiting subtours. This is sometimes called the *assignment problem* (AP). Next the solution is checked for any subtours, and if no subtours exist the solution is the optimal and the algorithm stops. If, however, subtours exist, the algorithm starts branching by selecting one of the subtours and adding constraints to the IP model that prohibits each arc in the subtour, one at a time. Each respective new model is solved and the feasibility of the solutions is checked. Once a feasible solution is found, the objective value, in this case the length of the tour, is saved as the new bound and the algorithm returns one level up in the branching and continues. The best solution is finally returned as the global optimum for the problem. [65] Intuitively one can see that solving the LPP is also possible using a similar algorithm. As the length of the longest possible path depends on the starting node of the path, the algorithm needs to iterate through a number of starting nodes before the longest path can be found. Not every node needs to be checked, due to the symmetry of the problem.

Part II
Overview of the Algorithm

The typical workflow of planning and delivering MR-HIFU treatments is described in Section 2.5.1, but a brief recapitulation will be given here. After the planning images have been acquired and imported into the MR-HIFU software, the planning process commences with the operator defining the planned target volume (PTV) in the stacks of images. Next, the operator defines one or more *clusters*, planes on which cells can be positioned. The planes can be positioned freely with three degrees of freedom, depth (anterior-posterior direction) and roll and pitch tilt angles. When filling the PTV with cells, the operator needs to decide not only how to position the clusters and where to position the cells in the clusters, but also how many cells to use and their sizes. Large treatment cells are more time-efficient than small cells, but they also cause more surface heating and require larger safety margins. For each individual cell that is placed in the PTV, the operator needs to verify a number of safety requirements in order to guarantee that the cell is safe to sonicate. Once all cells have been positioned, the operator needs to choose the power level to be used on each cell, relying on experience and expertise. If the operator wants to make adjustments to the treatment plan after the treatment has started, the process involves at least checking the safety of any new or moved cells and at most selecting cell size, choosing cluster and positioning the cell as well as checking its safety.

The problem with the current planning workflow is twofold: the planning process is very time consuming and the treatment is difficult to optimize. The time spent on planning the treatment is expensive as no treatment is delivered during this time. What is more, the planning phase and the safety checks keep the operator occupied with monotonous tasks instead of concentrating on fine tuning the treatment and on the well-being of the patient. The aim of the treatment is to maximize the ablated volume of the fibroid, without putting the patient at risk. On the other hand, not only economic reasons but also patient comfort requires that the treatment is delivered in the shortest possible time. A more comfortable treatment means that the patient is better able to lie still during the delivery. Each treatment is influenced by a vast amount of factors and parameters: the position and number of cells, cell size and form, power levels, heating and cooling times, sonication order, tissue parameters and structure *etc*. All of these factors, many of which are largely unknown prior of the treatment, are interrelated in a complex manner. The task of optimizing such a system manually, using only past experience and clinical expertise, is not feasible.

The solution for the problems is a (semi-)automatic treatment planning algorithm to improve the efficiency of the workflow and to help the operator plan and deliver better treatments. The aim with the development of a treatment planning algorithm is not only to free the operator from the monotonous task of filling the PTV with safe cells and checking the safety of cells added or moved by the operator, but also to optimize the treatment. By reducing the time spent on treatment planning the total treatment time is also reduced, and more time is allocated for treatment delivery. In order to be effective, the algorithm needs a good user interface which is not only easy to use but also versatile enough to be able to adapt to varied treatment cases. The algorithm should improve the workflow so that the operator feels that it adds value and prefers to use it over manual planning. The

algorithm should also attempt to find the optimal treatment parameters for each individual treatment case and produce an initial treatment plan that the operator can fine tune using expertise and experience. Furthermore, as the treatment is being delivered, the algorithm should be able to use the information gathered from delivered sonications as feedback to further optimize the initial plan or adapt it to changed conditions.

The realization of an effective treatment planning algorithm is therefore required to be easy to use and computationally relatively fast, at least faster than manual treatment planning. Furthermore, it needs to be flexible to be able to deal with varied and changing patients and treatment cases and preferably modular to be easily updated and improved upon as the mathematical modelling and computational methods improves. The mathematical optimization of the treatment requires that the anatomy of the target volume and its surroundings can be modelled to some level of accuracy in a geometric form, such as triangular meshes, and that all the objectives and constraints can be expressed in a mathematical form. At the same time, due to the complexity of a MR-HIFU treatment delivery as a system, some further assumptions and constraints regarding the problem are most likely needed. For example, by constraining the cell sizes and forms as well as the power levels to discrete, pre-defined sets of alternatives, the problem becomes more manageable and the algorithm fits well into the current SW framework.

An outline for an automatic treatment planning algorithm, which fulfils the requirements stated above, has been developed for this thesis. To the writer's knowledge, this is the first treatment planning algorithms for MR-HIFU that optimizes the treatment and has the ability to update the plan based on feedback.

This chapter starts with a presentation of the results of a product requirement elicitation process that was conducted in order to map the requirements for the algorithm. Next, a more elaborate description of the safety constraints margins and constraints is given. In the last section, an outline for the algorithm is presented.

7 Specifications and Requirements

The requirements for the algorithm were gathered through interviews with a clinical specialist, a product applications expert, a product manager, and a R&D physicist. The requirements defined through the elicitation process are presented below in Section 7.1. They are divided into four groups according to the nature of the respective requirement. The first group, *Outputs*, is concerned with the final outputs that the algorithm should provide. The second and third groups, *Ease of use* and *User interaction*, define how the operator interact with the algorithm using the Sonalleve MR-HIFU user interface, emphasizing the need of an easy-to-use but also flexible module that improves the planning workflow. The last group, *Technical requirements*, stipulates in further detail the technical details of the algorithm. Moreover, the requirements were separated according to their priority, *Must-have features* which are the bare minimum of a successful automatic planning functionality, and *Nice-to-have features* that would improve the performance of the algorithm but which are not vital.

In simple terms, the main constraint of an MR-HIFU treatment is the prolonged total treatment time, which should be minimized. On the other hand, the objective of the treat-

ment is to ablate as much of the tumour as possible. From these two mutually exclusive objectives, two different types of optimization problems were identified. First, the ablated volume should be maximized, given the constraint that the treatment time should not exceed a given time limit. As the sonications are delivered as well-defined cells, the problem is similar to the BMC and BUC problems discussed in Section 5.1.1 and 5.1.2 respectively. The second problem can be defined as the minimization of the total treatment time, given the constraint that at least a pre-defined portion of the fibroid is ablated. This problem resembles the MCCT and MCUCT problems described in Section 5.1.3. The identification of these two problems is also communicated in the requirements listed below.

The most important requirement for any clinical treatment is safe delivery. The safety requirements that the operator is required to evaluate for each individual cell, previewed in Section 2.5.1. In Section 7.2 the safety requirements are explained in further detail [30].

7.1 Requirements

Must-have-features are features required for an automatic treatment planning algorithm to be successful and improve on the treatment delivery workflow.

Must-have features

Outputs

R1 The algorithm should populate the PTV with a set of cells, of one or several sizes, which are (approximatively) optimal according to some default or user set criteria.

R2 The population of cells should be optimized in 3D, but the algorithm should also be able to optimize cell placements in planes (*clusters*).

R3 The algorithm should suggest in which order the cells should be sonicated, according to a default or user set strategy.

Ease of use

R4 The algorithm should create a therapy plan at its easiest by the push of button.

R5 Any manual fine tuning of the therapy plan done by the user should be as easy and intuitive as possible.

User interaction

R6 The user should be able to define the PTV manually, by for example segmenting the fibroid, or accepting the PTV produced by an automatic segmentation procedure.

R7 The user should be able to move, delete and add cells after the population has been produced. Moreover, after any alteration the user should be able to commence a re-optimization of the population using either the same or new optimization criteria.

R8 The user should be able to select the cell sizes to use, or let the algorithm decide.

R9 The user should be able to select the minimum separation or maximum overlap between the cells.

R10 The user should be able to choose to use a *conservative* or an *aggressive* sonication strategy, regarding in which order the cells are sonicated and how the cooling times are optimized. The conservative strategy would aim at minimizing the risk of NF overheating, while the aggressive strategy would aim at utilizing cumulative heating to further optimize the treatment.

R11 The user should be able to re-optimize the plan in the middle of the treatment, using the same or new treatment parameters (such as cell size), based on the feedback gathered from delivered sonications.

R12 The user should be able to manually change the default safety distances in the far field, as some patients are more sensitive to far field heating than others.

Technical requirements

R13 The first criterion of the optimization is to maximize the volume of the PTV that is ablated. The second priority is to minimize the treatment time.

R14 The algorithm should be capable of dealing with single fibroids or multiple smaller fibroids, and should suggest in which order the fibroids are to be treated for optimal results.

R15 The cells proposed by the algorithm should all be safe to sonicate. The algorithm should also be able to automatically evaluate the safety of cells altered or added by the user. All safety checks are done in 3D, regardless of if the cells are placed on a plane or in 3D.

R16 The cells positioned by the user should be static, meaning that they are included in the solution set even if removing them or repositioning them would improve the result, in the (re-)optimization process.

R17 The algorithm should be able to minimize the cooling times, and thus the total treatment time, and to further expand the total ablated volume by utilizing cumulative heating and taking into account the interaction of all sonications, not only successive ones.

Nice-to-have features are features that would add value to the performance of the algorithm but are not vital for the functionality of an early version of a treatment planning algorithm.

Nice-to-have features

User interactions

R18 The user should be able to define parts of the PTV with higher treatment priority, e.g. blood vessels.

Outputs

R19 The algorithm should be able to suggest optimal power levels for each cell, taking into consideration sonication depth, tissue parameters and other factors affecting the level of power needed to deliver an optimal cell size.

Technical requirements

R20 The algorithm should be able to estimate tissue parameters based on test sonications or realized treatment sonications, and utilize these parameters in the optimization of e.g. power levels or estimating near-field heating.

R21 The algorithm should be able to utilize information about delivered dose and changes in tissue parameters as feedback to re-optimize the treatment plan.

R22 The algorithm should be able to further optimize the treatment time and ablated volume utilizing cumulative heating by altering the sonication times and power.

During the interviews it was realized that an effective auto-segmentation procedure is crucial for the success of an auto-population algorithm. Not only would it be used for OAR detection and thus facilitate the safety checks, but it would also improve on the time required by the user to produce an accurate PTV over the fibroid. Moreover, it was speculated that in order to accurately optimize the cooling times and utilizing cumulative heating from the whole set of sonications, a coarse 3D BHE-equation solver would be needed, which in turn would require an approximative anatomical model of the target volume.

As there is an obvious need to let the user update the plan as the treatment evolves, the optimization process also needs to be computationally fast. This requirement implies that the accuracy of the mathematical models used and the required computational effort need to be balanced.

7.2 Safety Considerations

Several safety factors need to be taken into consideration when planning a set of cells to be sonicated. Currently the safety of each cell is checked manually, one at a time, by the operator. This can be time consuming, despite the fact that the Sonalleve MR-HIFU software does facilitate the process by providing various graphical overlays on top of the planning images of the patient. Below are presented the primary safety concerns regarding uterine fibroid HIFU sonications, and the way the safety margins have been defined in order to ensure safe sonications, based on the official Philips MR-HIFU training material [30].

OARs in the near and far field In UF applications, OARs in the NF and FF include, among others, the bowel and the spine. In order to ensure that the ultrasound energy delivered to OARs in the NF is kept at safe levels, it is stipulated that no part of an OAR can be located inside the US beam between the transducer and the focus point. In the FF, OARs may be located inside the beam but only at a safe distance from the focus point. The safety distance depends on the cell size and cell type, and varies in the range 40-60 mm.

Uterine serosa The uterine serosa, also known as the *perimetrium* or simply the *serosa* in the context of uterine fibroid ablation, is a smooth membrane that secretes a lubricating serous fluid, which reduces friction from muscle movement between organs. The serosa encloses the uterus, and is thus the outermost layer of the uterus, which in turn means that it can be located quite close the fibroid. In order to avoid thermal damage to the serosa, cells should be positioned at a safe distance from the serosa. The safety margin is cylindrical, with its long axis parallel with the cells long axis, centred at the focus point, and with dimensions dependent on cell size and cell type. Typical values for the height are 50-80 mm and for the radius 15-25 mm.

Heat build-up in the near field If several cells are sonicated consecutively without sufficient cooling times in between, heat build up in the NF might lead to skin burns or discomfort for the patient. As the MR thermometry is only able to measure relative temperature changes, it is not able to monitor heat build up from consecutive sonications, without continuously scanning the target volume. The only guidelines given to minimize the risk of excessive NF heating are to use long cooling times and to minimize the overlap and maximize the distance between successive sonications areas of NF heating.

Energy density in the near field If a large amount of power is used to sonicate a cell positioned close to the skin, the heat build up from one single sonication may be enough to cause skin burns. Even though the MR-HIFU software monitors the NF heating during the sonications, this is an unnecessary risk which should be avoided. The software also issues a warning when the operator attempts to use a power level that the SW deems is too high considering the sonication depth. A safe level of power is highly dependent on tissue parameters such as perfusion and absorption, on the local tissue structure, as well as the size of the intersection area between the conical beam and the NF plane.

8 Suggested Outline

Treatment planning algorithms have been developed before, especially in radiotherapy, as discussed in Section 4.1.1. In treatment planning algorithms for brachytherapy, there are two different approaches: expert systems, relying on databases of earlier successful cases and learning neural networks, and mathematical modelling and optimization, based on effective optimization algorithms and computational power.

Expert systems have proven their effectiveness and speed in medical diagnostics and HDR treatment planning, but an implementation of one would require a large amount of various tumour cases and successful plans to train the system. As MR-HIFU is still a relatively young modality, there is no large pool of delivered treatments available. Even though the Sonalleve software does save an extensive copy of every treatment, segmenting these cases would either be very time consuming or require an automated segmentation algorithm. What is more, even though the system would be able to produce a plan based on past experiences it would still require a safety check capability to be able to ensure the safety of the treatment. Despite this, using expert systems for HIFU treatment planning is certainly an interesting thought.

HDR treatment planning based on a mathematical algorithm generally starts with a segmentation procedure. The segmented target holds the candidate seed points. Calculation points are distributed inside and on the surface of the target, the surrounding normal tissue, and the OARs. Using constraints on delivered doses and a model of how the dose spreads, the HDR optimization problem often consists of solving a mixed integer programming problem. The MIP model is able to solve not only the optimal choice of seed points, but also the dwell time of each the seed, simultaneously. There are some important differences between HDR and HIFU, which influence the way a treatment planning algorithm can be implemented:

- The spread of dose in HIFU is not as simple as for HDR, but a complex function of heat delivery, dissipation, time, and several other factors.
- The dose spread in HIFU is more locally constrained, and can be confined to a very small volume.
- There are considerably more parameters involved in the delivery of a HIFU treatment than in HDR: cooling and heating time, sonication orders, power levels *etc.*.
- A HDR treatment is not limited by the physical constraints of the apparatus.
- In HIFU the safety considerations are more numerous and are more complicated to formulate in a linear programming model.
- The seed points in HDR are constrained by the discrete positions of the catheters, which is not the case with HIFU.

These differences imply that the MR-HIFU algorithm, in practice, needs to be split up in several stages to be able to perform the same task as the HDR algorithms does in a single step. Even though it would theoretically be possible to balance all factors affecting the outcome of a HIFU treatment, the problem would very quickly grow infeasible to solve. Some

further constraints and assumptions are, therefore, needed to keep the problem size manageable. Finding a set of allowed cell positions, optimizing the subset of chosen cells and the optimization of sonication parameters need to be separated into individual procedures, which communicate their results forward to the next procedure. Basically, separating the optimization process into sequential steps means that the problem loses some generality, and the solutions might not be the global optimum. In practice, this loss of generality should not affect a conservative sonication strategy, but an aggressive strategy could be influenced somewhat. On the other hand, the workflow represents the most intuitive flow of information in the planning process and replicates the way the operator currently plans the treatments. An intuitive workflow is required when the operator is given an option to intervene and issue re-optimization of the whole or only parts of the plan.

The suggested outline for a MR-HIFU automatic treatment planning algorithm is presented in the form of a flowchart in Figure 5. The flowchart bears resemblance to the general outline of HDR treatment planning algorithms, starting with a segmentation procedure and continuing with an optimization procedure, but is adapted to the special requirements of MR-HIFU. Each individual step in the optimization process is allocated to a separate module: finding feasible cell positions in the *initialization module*, selecting the optimal subset of cells in the *population module*, and optimizing the sonication parameters in the *sonication parameter module*. The presented structure involves an intuitive flow of information through the whole procedure, and resembles the workflow currently performed manually by the operator. Moreover, the modular structure of the algorithm introduces flexibility to the sequential improvement of parts of the algorithm. Another important feature of the algorithm is the possibility to use feedback from the delivered sonications to improve on the original plan, a feature that is not available in HDR treatment planning. The modules are presented in detail in Sections 8.1 – 8.4.

The only constraint added to the problem is that the sonications are delivered as well-defined cells. This is the way cells are currently delivered by the Phillips MR-HIFU system and is also how the users have learnt to deliver doses. Delivering dose as well-confined cells implies that the optimization problem of selecting the best subset of cells that cover the maximum amount of volume inside the target volume is not unlike the covering problems familiar from network optimization, BUC and MCUCT. These two problems also transfer naturally to the HIFU environment; the cost associated with each cell is the time it requires, notably the heating and cooling time, and the profit is the weighted sum of covered target points (TP). The target points are a set of artificially created points inside the target, which are used to discretize the target volume. The constraint on cell forms not only allows the utilization of algorithms proven successful in network optimization but it also implies that the algorithms fit well into the current Sonalleve software framework. What is more, a number of the modules developed for the algorithm could also be applied to refine plans produced by expert systems and help manual planning.

The alternative to using constrained cells to deliver doses would be to use more complicated sonications paths, producing more exotic coverage patterns. This alternative would offer more flexibility to the treatment as the algorithm would not be constrained to optimize ellipsoidal cells inside the target volume. The dose development in living tissue is, however, a complex process and the accurate control of more diverse sonication paths than circles is not a trivial matter. This can be seen in that the Sonalleve MR-HIFU software

currently only allows the use of cells for dose delivery, and as the current safety checks are based on the assumption of simple cell sonications, more complex patterns would require other types of safety checks. What is more, the range of the electric deflection available with the current transducers is quite narrow, which limits the possibilities of sonication along more complex paths.

The following sections introduce the modules of the automatic population algorithm in more detail. Even though it is assumed in the discussion that the algorithm is for planning a treatment for uterine fibroids, as it is the main application for the Philips MR-HIFU system at the moment, the algorithm is designed to be general enough to be applied to any MR-HIFU ablation application.

8.1 Preparatory Steps

The first, preparatory steps of the treatment planning workflow consist of three separate modules: a segmentation module, a module representing the HIFU software framework, and a module conducting tissue parameter approximation.

The segmentation module takes as input the planning images from the scanner. The images are analysed by the module, and the anatomical regions of interest are discretized as sets of points. In particular the fibroid, the serosa, OARs in the NF and FF, as well as any areas of the target with pronounced importance for the delivery of the treatment, such as the blood vessels of the fibroid, are discretized. These point sets are then passed on to the initialization module (see Section 8.2), which produce triangular meshes of the ROI based on the point sets. The segmentation can be performed either manually or preferably automatically. The result of the segmentation is presented in the GUI, giving the operator a possibility to fine tune the segmentation before the result is passed to the next module. As the geometrical representation of the targeted tissue is required for the safety checks, the segmentation process is also a vital part in the functionality of the whole treatment planning procedure.

The MR-HIFU SW framework provides the initialization module with important systems parameters, such as available cell sizes and forms and safety parameters and margins. What is more, the framework provides a model of the ATA, which is essential for determining the volume the transducer is able to sonicate.

The tissue parameter approximation module outputs approximative values for important tissue parameters, such as diffusion and perfusion in target volume, to the initialization module. Research into parameter approximation, using test sonications, has been reported by Dragonou *et al.* [10]. These parameters can then be used to more accurately estimate the power needed to deliver a cell at a given position. The power level can, in turn, be used to give an estimate of the cooling time needed after the sonication. Thus, it also gives the time cost of the sonication. Similarly, the power level estimations can be used for the NF energy density safety checks. The tissue parameters can also be used by the sonication parameters module for heating and cooling time, power level, and sonication order optimization. Tissue parameter approximation is not vital for the success of the treatment planning, but would be a considerable asset to the planning algorithm.

8.2 Initialization

The initialization module outputs a list of positions, including x-, y-, and z-coordinates as well as angulations (roll and pitch angles), accompanied by the set of cells that can be sonicated at that position. The module also provides a set of target points with possible weights. Finding the set of candidate positions is a three step process. First, the module creates triangular meshes of the target volume, the serosa and the OARs in the NF and FF using the point sets provided by the segmentation module. Triangular meshes are required for the safety and feasibility checks of the cell positions. Second, the module starts filling the target volume with candidate positions, with individually defined accuracies in each of the five dimensions and using the cell sizes and other parameters defined by the user. A position is labelled *feasible* if it fulfils all of the following conditions:

- The position is reachable by the transducer
- The position is inside the target volume
- The position fulfils all safety requirements, described in Section 7.2, for one or several different cell forms and/or sizes.

The safety checks utilize the newly created triangulation meshes together with the parametric data provided by the HIFU SW framework.

Next, the module estimates time costs for each position-cell combination. The estimation can be based on any factor that influences the time cost associated with the cell at the given position, but as the NF heating is the primary reason for prolonged cooling times in the UF case, it is natural to base the time cost estimate on the amount of NF heating the cell creates using a given power level. The estimates can be improved by using output from the tissue parameter approximation module.

The module also creates a set of target points. The only requirement for the target points is that they are inside the target volume. If the user has defined a partition of the target volume with special importance, the initialization module can provide a set of weights which emphasizes the target points in this region more. Finally the set of feasible positions-cell combinations and their costs, together with a set of target points with weights, are communicated to the next phase in the process, the population module.

8.3 Population

The population module produces the optimal set of cells, or an approximation thereof, for the treatment. In this context the term *optimal* signifies a set of cells that has been produced by an approximation algorithm, and not specifically the mathematically optimum set. An approximation algorithm aims to produce solutions close to the mathematical optimum to problems that cannot be solved exactly in reasonable time, but do not guarantee that the found solution is the global optimum. The set of feasible cells and target points together with the costs and weights of the cells and target points, respectively, are provided by the initialization module. Using these sets, the module initiates an optimization algorithm which solves either the BUC or the MCUCT problem, as described in Sections 5.1.2 and 5.1.3, respectively. As noted earlier, the BUC problem asks for the set of cells that cover the

maximum weighted amount of target points, with the constraints that no cells can overlap and the total time cost for the set of cells does not exceed a budget, provided by the user or defined by the system. In contrast, the MCUCT problem asks for the set of cells with the minimum total cost provided that no cells overlap, and that the coverage or weighted coverage exceeds a user or system defined threshold. If the user chooses to allow a certain amount of overlap or requires a given amount of gap between the cells, the size of cells fed to the optimization algorithm is decreased or increased accordingly, so that the BUC and MCUCT algorithms are able to handle the problem.

The user is also given the choice of selecting or positioning a number of cells manually, before initiating the optimization algorithm. These cells are regarded as static by the algorithm, meaning that they are always included in the final solution even if the solution would improve by removing or repositioning the cells. The only requirement on the user-defined cells is that they are safe and feasible, which the module checks in a similar manner as in the initialization module.

As discussed in Section 5, both the BUC and the MCUCT problems are *NP*-hard. This means that, in theory, no algorithm is able to solve them exactly in polynomial time. For this reason, the algorithms used for solving the problems are only approximative. If the problem is small, also deterministic algorithms, which solve the problems exactly, might be able to find a solution in reasonable time.

The module is fed feedback after every sonication in the form of delivered doses. The user has the option to update and refine the treatment based on the received feedback. The delivered doses are simply regarded as static cells by the optimization algorithm, which means that no alterations need to be made to the algorithm to take the delivered doses into account.

8.4 Sonication Parameters

The sonication parameters module receives a set of cells from the population module and attempts to optimize the heating and cooling times of each cell and the order in which the cells are sonicated. The optimization is driven by the user's choice of sonication strategy. A *conservative* strategy aims to minimize the risks of delivering extensive amounts of ultrasound energy in a short period of time, such as overheating in the near field, while keeping the total treatment time at a reasonable level. An *aggressive* strategy, on the other hand, aims to deliver the treatment as quickly as possible and gain advantage of the cumulative heating in the target volume to further expand the ablated volume.

When using a conservative strategy, the sonication order is optimized by minimizing the overlap and maximizing the distance between the areas of NF heating in two consecutive sonications, as described in Section 3. The area of NF heating is the intersection areas of the sonication beam and the NF plane. This problem resembles the LPP, presented in Section 5.2. The LPP is *NP*-hard and therefore approximative algorithms are needed for finding good solutions for larger instances of the problems. For smaller problems BBs can prove effective at finding the global optimum of the problem.

Optimizing an aggressive strategy might prove more difficult, as utilizing the cumulative heating in the target volume requires simulation of the heat development, as shown by Mougenot *et al.* [26]. In order to produce usable simulation results, the BHE needs a

coarse anatomic model with approximative tissue parameters. These are fed to the module by the tissue parameter approximation module, and are refined as more sonications are delivered. Utilizing cumulative heating would, however, introduce the need for further safety requirements, such as restricting the heat spread and temperature rise outside the target. Optimizing by using such constraints are common in brachytherapy, as discussed in Section 4.1, but while the radiation dose delivered to a given point is only a function of the elapsed time, the distance to the source and the activity of the source, the heat spread is considerably more complex to calculate. As Payne [24] has shown, optimization of cooling and heating times can also be performed in a similar manner, by solving a 3D BHE for a simplified anatomic model, regardless of the chosen sonication strategy.

8.5 Feedback

After the optimized plan has been produced by the algorithm and the first cells of the treatment have been delivered, the algorithm receives feedback based on the outcome of the sonications. This data is then used to update and to refine the plan. Of primary interest for the algorithm is information about the volume and form of the delivered dose, the amount of NF heating, heat diffusion and cooling times as well as updates to the ATA caused by alteration in electronic corrections made to the focus.

The initialization module is fed information about any updates to the ATA model and new approximations of tissue parameters. The updated tissue parameters are used to improve the NF heating approximations, which in turn affect the time cost of the cells. Moreover, they can also affect the approval or disapproval of the cells. This, in addition to the updated ATA model, can affect the set of feasible cells that is passed to the population module. Also, the user is given the possibility to change treatment parameters, such as cell sizes used and FF safety distances, which might render a re-optimization of the plan necessary.

The population module uses the updated set of feasible cells to update the set of chosen cells through a re-initialization of the optimization algorithm. What is more, the algorithm uses the reported size and locations of the delivered doses to restrict the available space in the target volume. From the point of view of the algorithm, these delivered doses are handled as static cells in that they must be part of the solution produced by the algorithm. In this way, there is no need to alter the algorithm for it to be able to take the delivered doses into account. The importance of refining the plan based on the realised size of the dose grows when the dose is much smaller or larger than planned, for example due to premature termination of the sonication of poorly dimensioned power levels.

The sonication parameter module re-optimizes the new set of chosen cell received from the population module. The module also uses the updated tissue parameters to refine the sonication parameters using an updated anatomic model of the target volume. Moreover, information about smaller or larger than planned treatment cells indicates that the modules power level estimation is working sub-optimally and the feedback is used to correct the estimations made about the remaining power levels of the cells.

These re-optimizations using feedback should only be initialized by the choice of the operator. Preferably the user interface could give the operator a notice that an improvement to the plan is possible, and the system should warn the user about continuing with an infeasible or unsafe plan.

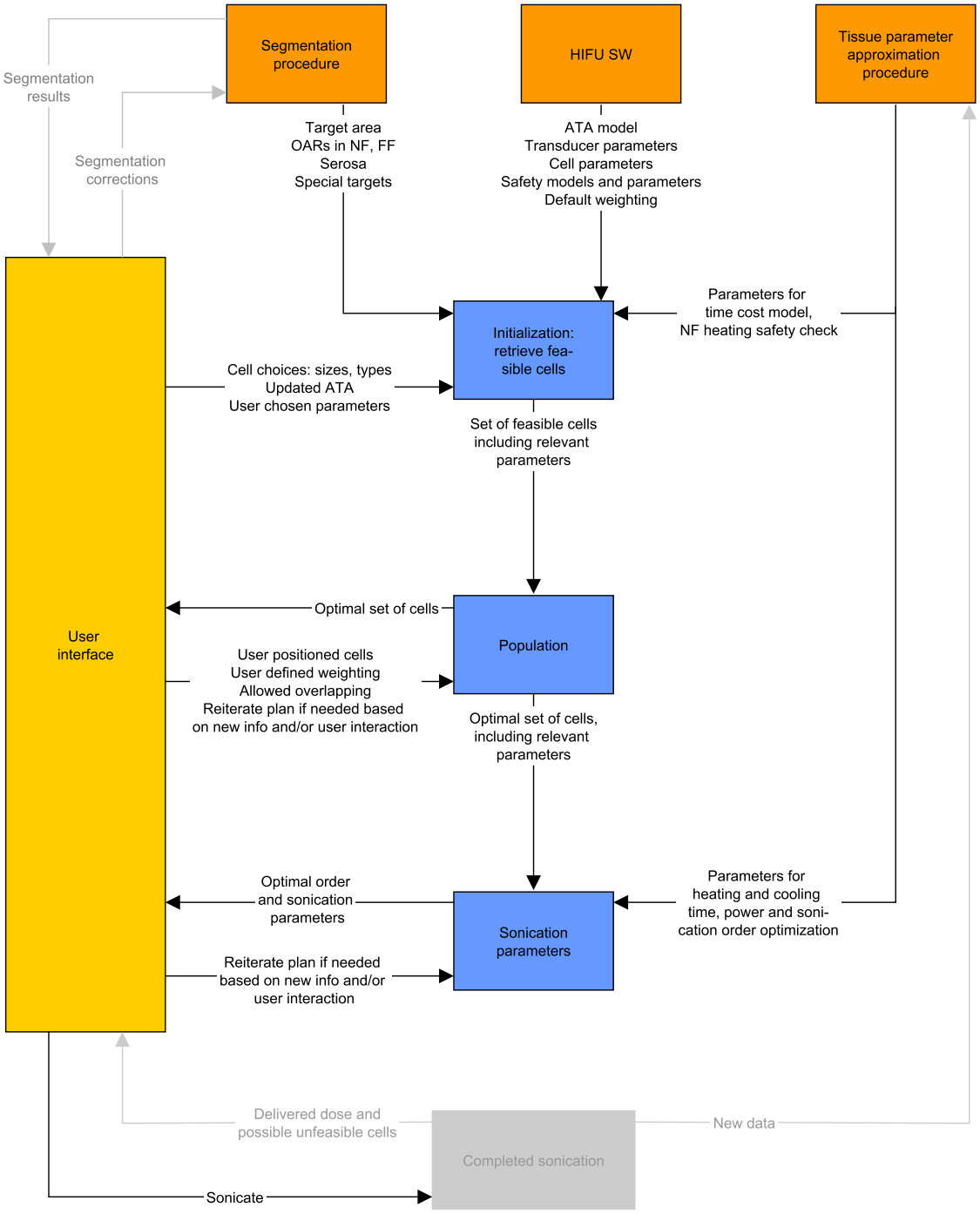


Figure 5: Suggested outline of the automatic treatment planning algorithm for MR-HIFU.

Part III
Implementation of the Algorithm

The implementation of a prototype of an automatic treatment planning algorithm was conducted as a feasibility study, without trying to produce a new functional part of the Sonalleve MR-HIFU software. The emphasis of the study laid on implementing a simple but functioning entity and evaluating alternative optimization algorithms.

The resulting algorithm is able to produce off-line treatment plans for actual clinical cases taking into account the transducer ATA and most of the safety consideration presented in Section 7.2, populate cells in an approximately optimal manner in clusters or in 3D, and optimize the sonication order of the optimized cells in accordance with a conservative sonication strategy. The optimization of an aggressive sonication strategy was not implemented. The user interface, segmentation modules and tissue parameter approximation algorithms are beyond the scope of this thesis. To summarize, the presented algorithm fulfils or has the capability to fulfil all the *Must-have-requirements* presented in Section 7 that does not rely on cumulative heating or optimization of cooling and heating times or power levels (R17, R19, R20).

The chapter is structured as follows. First, a review is given of how the modules of the algorithm were implemented. Second, the different algorithms used in the population module and in the sonication order optimization module are compared in a series of artificial test cases. Next, a real clinical case is used to demonstrate the feasibility of the initialization and population modules and compare the implemented coverage optimization algorithms. A number of produced treatment plans are also presented. Lastly, the results are reflected upon.

9 Description of the Modules

Given below is a description of the implemented prototype of a functioning automatic treatment planning algorithm. Where appropriate, pseudo code of the modules is presented.

All code was produced using IDL 8.0 (ITT Visual Information Solutions, 2010), except for the linear programming solvers used for the BB algorithms, which were implemented as GNU MathProg modelling language models and solved using the GNU Linear Programming Kit Solver, ver. 4.46.

9.1 Initialization Module

The patient's anatomy in the region of interest is transformed by segmentation into a format that the mathematical models can understand. The implemented prototype utilizes a segmentation procedure which positions discrete points, slice wise, on the surface of the object, based on the acquired planning images. These points are passed to the initialization module, which transforms them into triangular meshes using a meshing procedure. This

procedure plays a crucial part in the functionality of the module and the rest of the algorithm, as all safety checks require a mesh of the anatomic region. The choice of accuracy of the mesh is a balancing act between anatomic realism and computational efficiency; more triangles are able to represent the ROI more realistically but will unavoidably render the safety checks computationally heavier. The implemented algorithm was inspired by Liu *et al.* [85], who presented a meshing algorithm designed to create meshes based on point sets in which the points are positioned in layers or slices. This suits point clouds from segmented MRI scans well, as they are in general arranged in layers. The algorithm is fast and functional in the respect that it is able to create closed surfaces, but encounters problems when the transition between the points in consecutive slices is large. Further details on the algorithm can be found in [85]. The procedure creates meshes of the fibroid, the serosa, and the OARs in NF and FF, based on the input received from the segmentation module.

The next step in the initialization module is to create a set of feasible cell positions. For a cell to be feasible, it needs to fulfil the list of requirements presented in Section 8.2. The set of feasible points is created by iterating through a set of possible positions created in and around the target volume and checking them against all requirements. The candidate positions can be spread out in 3D or in clusters, but the clusters need to be defined by the user as the algorithm does not optimize their position or angulation. The largest cell size, if any, which is allowed at each respective position is stored along with the positions.

Determining the feasibility of a position is a five-step process. First, the procedure checks whether the position is inside the fibroid or not. This is done by calculating the solid angles from the given point to all triangles in the mesh, as it is known that if a point is inside a closed surface the solid angle to the whole surface is exactly 4π and zero if it is outside the surface [86]. The solid angle is calculated using the method presented by van Oosterom and Strackee [87]. Next, the procedure evaluates whether the transducer is able to reach the position, if it is inside the transducers ATA. In this prototype the ATA is calculated using a simple model of the positioner, implemented in IDL, which has been verified to replicate the actual ATA calculations done by the MR-HIFU SW well.

If the cell position is both inside the fibroid and reachable by the transducer, safety checks are carried out. The safety requirement for the serosa is verified by calculating the shortest perpendicular distance from the axis of the cylinder that defines the safety margin to a point in the serosa mesh. This distance is then compared with the safety margin, and if it is greater than the radius of the cylinder the safety check is passed. The largest cell that is allowed at the position is recorded.

In the NF, no OAR can intersect the conical US beam. The procedure for the NF safety check determines whether any of the triangles intersect the US cone, using a method proposed by Eberly [88]. The FF safety check is somewhat more complicated as the distance between the focus point and any point in the OAR intersecting the beam needs to exceed a cell specific safety limit. In practice, this requires a method that computes the distance from the cells centre to a point on a mesh triangle in the OAR. Such a method was developed for the module and is presented in Appendix B. Again, as the safety margin is larger for larger cells, the largest cell that is allowed at the position is recorded.

If a position passes all safety checks it is labelled *feasible* and saved together with cell sizes that are allowed at that position. Noticeably, many of the safety checks are based on computing the same calculations for all or a large portion of the triangles in the meshes. For

this reason, reducing the number of triangles in the mesh is an effective way of improving the running time of the computations.

For clarity the whole procedure of retrieving a set of feasible positions is presented as pseudo code below.

Algorithm 1 `RetrieveFeasiblePositions`

Input: Mesh for target, serosa, NF, and FF, safety parameters, ATA model
Output: Set of feasible cells

```
Load cell specific form and safety parameters
Create a set of positions which are inside the target
for all positions do
  if the position are inside the ATA then
    Find the largest possible cell that can be placed at the position without being too close to the serosa
    if The largest cell is not 0 mm then
      if There is no OAR in the ultrasound beam in the NF then
        Find the largest possible cell that can be placed at the position without an OAR being too close in the FF
        if The largest cell is not 0 mm then
          Save the position as feasible together with the min(cell size from serosa, cell size from FF)
        end if
      end if
    end if
  end if
end for
return Feasible positions and the largest allowed cell at the respective position
```

The NF energy density safety check was not implemented in the procedure because, at the moment, it is not mentioned in the Philips training material as one of the safety checks to be performed [30].

The initialization procedure also creates the set of target points. The only requirement on the target points is that they are inside the target. In practice, this means that cells positioned on the edge of the target provide less coverage than cells in the middle. Also, if the user chooses to prioritize a certain part of the target volume above the rest, the module produces a list of the weightings to accompany the set of target points. Lastly, the module assigns a time cost to all cell-position combinations. The cost should reflect the time required to sonicate the cell and the preceding cooling time. This means that, in order to achieve a good estimate of the cost, one would need to simulate or in some other way estimate the NF heating caused by sonication the cell using an optimal level of power. As NF heating is a complex matter, and the choice of optimal power levels even more so, the implemented cost setter simply assumes that deeper positions (in the AP-direction) require more power. This, in turn, causes more NF heating and, consequently, requires proportionally more cooling time. This assumption is only partly right, as the size of the area of

the intersection between the US cone and the NF plane also influences the NF heating. Positions closer to the skin have smaller intersection areas, which leads to more concentrated heat deposition in the NF as well as easier heat diffusion from the focus point towards the skin, both causing more NF heating.

9.2 Population Module

The population module optimizes the subset of chosen cells that either cover the maximum amount of volume uniquely, so that the budget constraint is not exceeded (the BUC problem, see Section 5.1.2), or minimize the total cost, so that the unique cover exceeds a set threshold (the MCUCT problem, see Section 5.1.3).

The first step in the population module is creating the so called *coverage matrix*. This binary matrix communicates which target point can potentially be covered by which cells, and *vice versa*. Cells centred at the same point in space may cover different target points, due to differing sizes and orientations. The cells are assigned to the columns, and the target points to the rows. For each target point that is inside a given cell a 1 is assigned to the corresponding row in the coverage matrix. An example of a simple coverage matrix is shown below.

$$\mathbf{C} = \overbrace{\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}}^{\text{Cells}} \Bigg\} \text{Target points}$$

In the example, the first cell (first column) covers target points two, three and five, and so on. The coverage matrix is created by iterating through every cell and determining which target points are located inside the cell, based on the cell sizes and models that software framework provides. For simple cell forms, such as ellipsoids, the evaluation can be performed analytically, but for more complex forms a method based on solid angle calculations could be implemented. After the coverage of all cells has been established, target points that are not covered can be removed from the set to decrease the memory load.

According to the requirements presented in Section 7, the user should, however, have the possibility to select the overlap or gap between cells. Algorithms designed to solve the BUC and MCUCT problems do not take this into account. For this reason, the coverage matrix procedure takes as input the users desired overlap and reduces or increases the sizes of all cells by a corresponding amount before their coverage is determined. This allows the introduction of a user defined overlap parameter into the algorithm without altering the structure of the optimization algorithms.

After the coverage matrix has been created it is passed on to the population algorithm. The general guideline for MR-HIFU operators is to start filling the volume or cluster using the largest cells possible, and then reduce the size gradually once no more large cells can be added. From the point of view of the population algorithm, it is not trivial to choose how to

fill cells of different sizes. The choice lies between letting the algorithm fill one size at a time, starting form the largest size and then gradually reduce the size, or attempt to optimize by using cells of all sizes simultaneously. Splitting up the optimization problems into steps means that some of the generality is lost, meaning that the two approaches will not have the same global optimum. Both approaches were implemented, in order to compare their performance and how the loss of generality influences the results.

The complete population module is presented as pseudo code below.

Algorithm 2 `PopulationAlgorithm`

Input: Set of feasible cells, Set of target points, Cell costs, Target point weights, Cell models, Overlap margin
Output: Subset of chosen cells
Reduce/increase the size of each cell by the overlap margin
Create the coverage matrix by evaluating which target points are inside which cell
Choose the optimal set of cells by optimizing one size at a time or all sizes simultaneously
return Subset of chosen cells

All in all, four different population algorithms were implemented, all of which are introduced in Section 6. The algorithms were chosen for their varied *modus operandi*: one is deterministic (BB), one is guided stochastic (GA), one is simple and quick (greedy), and one is a combination of random selection and deterministic local search (GRASP). Moreover, all algorithms have been used in network optimization problems with proven performance. A more detailed description of the implemented algorithms will be given below, and pseudo code for each of the algorithms can be found in Appendix C.

The greedy algorithm used for solving the BUC problem is separated into two different heuristics, as in [62]. The first heuristics selects the cell with the highest weighted coverage-to-cost ratio from the subset of cells that will not cause the time budget to be exceeded or cause an overlap with any of the cells that have already been chosen. If more than one cell has the highest ratio, the heuristic selects one of them at random. The heuristics continues adding cells to the subset of chosen cells until no more cells can be added, either because the time budget or the overlap constraints would not be satisfied any more. The second heuristics functions in much the same way, with the difference that it selects at each step the cell with the highest weighted coverage. The greedy algorithm for the MCUCT problem resembles the BUC heuristics, with a few important differences. The algorithm selects the cell with the lowest cost-to-weighted coverage ratio, and keeps selecting cells until the profit threshold is exceeded. If the heuristic encounters a situation where no more cells can be added while the coverage threshold is yet to be satisfied, the algorithm fails.

The GRASP algorithm is based on the greedy algorithm, but instead of always choosing the best candidate, the algorithm chooses one out of many good candidates at random. The good candidates are the subset of cells that have a goodness value $\beta * 100\%$ of the best goodness value. Here the goodness value may be weighted coverage-to-cost ratio, weighted coverage, or cost-to-weighted coverage ratio. The decision parameter β is set by the user or the system between 0 and 1. For $\beta = 0$ the algorithm simply chooses a cell at random, for $\beta = 1$ the algorithm reduces to a greedy algorithm. After the candidate has

been created, a local search is initiated. The local search iterates through all the selected cells and exchanges them for each of the not chosen cells, before evaluating the goodness and feasibility of the new subset. If the new subset is better than the currently best one, it is recorded as being the new currently best candidate. These two procedures are repeated a number of times, and the best solution from those iterations is returned as the final solution. The number of iterations is determined by the user or the system. The BUC version of the GRASP algorithm uses a heuristics with the same functionality as the greedy BUC heuristic, and the MCUCT version uses similar logic as the greedy MCUCT heuristic.

The implemented genetic algorithm has all the basic elements of the genetic algorithm presented in Section 6.2, improved with a number of heuristics. Genetic algorithm strives to maximize the fitness, so the fitness values of the problems needs to be formulated as maximization problems. The fitness value for the BUC problem is simply the total profit of the chromosome, while for the MCUCT problem, being a minimization problem, the reciprocal of the chromosomes total cost was used as the fitness value. Only roulette wheel selection was implemented. The fitness values were scaled using sigma scaling to normalize the selection pressure throughout the evolution of the process. Furthermore, the roulette wheel selection was augmented with SUS, which has been shown to improve the selection process for smaller populations. Further information about the selection operators, scaling and SUS can be found in Section 6.2.1. In order to further steer the development of the population in the direction of feasible solutions, elitism was used to ensure that the best chromosomes are automatically included in the next generation. This way a good solution is never lost or destroyed by the crossover or mutation operators. Uniform crossover and normal random mutation was used, together with an operator that removes identical chromosomes through mutation in order to preserve the diversity. In order to assumedly improve the performance, the algorithm was given a good start using either of two different seeding methods. The first population could be created either so that one chromosome is created using the greedy algorithm while the rest is randomized or so that the whole population is created using a GRASP like algorithm, without the local search procedure. This way the GA is given a good starting point, which assumedly will help it to find good feasible solutions in less iterations.

The budget and threshold constraints and especially the overlap constraints are challenging for the GA. Developing closed operators for coverage problems with overlapping constraints proved very difficult, and therefore the algorithms needed constraint handling. In GAs, constraints are generally dealt with either by using penalties, reducing the fitness value of the infeasible chromosomes in some way, or by using heuristics repair operators, which create feasible chromosomes out of infeasible ones. For comparison, both methods were implemented. The infeasibility penalties were realized simply by assigning the fitness value of a chromosome that did not fulfil all constraints to zero. The implemented repair function for the BUC problems iterates through the population and removes cells from the chromosome until both the budget and the overlap constraint are satisfied. The MCUCT repair function first removes cells until the overlap criteria is satisfied, and then adds cells that do not cause overlap until the threshold criterion is satisfied. Even though these iterative processes are performed numerous times per generation and can thus prove time consuming, it does guarantee that every chromosome in the population is feasible. To reduce the computation time needed, the user can choose to initialize the repair operator,

for example, only every tenth or hundredth generation.

Even though multi-objective optimization is possible using genetic algorithms [43–45, 48, 50, 52] and even though objectives of the coverage maximization (BUC) and cost minimization (MCUCT) could be combined into one multi-objective problem, it is not trivial to determine how the two objectives should be weighted. Therefore, multi-objective GAs were not investigated further in this thesis.

As IDL lacks efficient linear programming solvers that are able to solve larger problems or integer programming problems, the GNU Linear Programming Kit (GLPK) Solver was used to solve all relevant IP problems. The IP problems (5.2) and (5.3)(with the added unique coverage constraint as described by Equation (5.2c)), were modelled using the GNU MathProg modelling language and passed on to the GLPK Solver via IDL.

9.3 Sonication Parameter Module

The implemented sonication parameter module only allows the optimization of a conservative sonication strategy. The reason for this is that it is not clear how an aggressive strategy, utilizing cumulative heating as well as exotic heating and cooling times and power levels, should be optimized. Furthermore, based on literature regarding cumulative heating and cooling time optimization [24, 26] it seems probable that the method would require several occurrences of a simplified BHE to be solved. This might get computationally very demanding even for small problems. Moreover, no research has been published on the subject of ordering cells optimally for utilization of cumulative heating. For similar reasons, no cooling or heating time or power level optimization procedures were implemented in the prototype of the module.

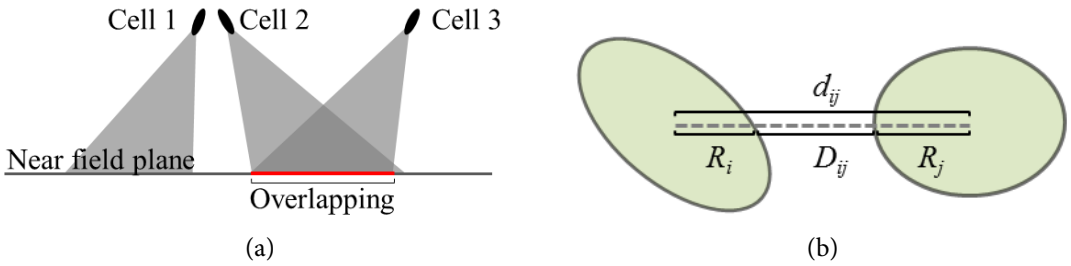


Figure 6: (a) demonstrates a situation in which tilted cells cause the US beams of two sonications to overlap, even though the centres of the cells are far apart. In this case, to minimize the risk over overheating in the NF, the cells should e.g. be sonicated in the order Cell 2, Cell 1, and Cell2. (b) illustrates how the distances between intersection ellipses are calculated, as in Equation (9.1).

The prototype module is able to optimize the sonication order in two ways, both based on solving the LPP. The first method maximizes the distance between the centres of the consecutive sonication cells. The distance between the centres is calculated as the Euclidean distance between them. This method is, therefore, quite simple and does not contribute with much more computational burden to the problem in addition to solving the LPP. The second method is slightly more complex, and aims at maximizing the distance

between the centres of the areas of NF heating from consecutive sonications. In practice, this implies a maximization of the distance between the ellipses formed in the intersection between the NF plane and the conical US beams of consecutive sonications. The former method is computationally less demanding, and works well when all cells are positioned in the same cluster. If the population, on the other hand, includes cells that have differing angulations, the latter method is required to take into consideration how the beams intersect with the NF plane, forming elliptic curves of intersection. An example of how cells with differing angulations can lead to overlapping in the NF is shown in Figure 6a. The distance between the intersection ellipses is calculated as

$$D_{ij} = d_{ij} - R_{ij} - R_{ji}, \tag{9.1}$$

where D_{ij} is the calculated distance between the two ellipses, d_{ij} is the distance between the ellipses centres, and R_ij and R_ji are the *radii* of ellipse i and ellipse j in the direction towards the centre of the other ellipse. The distance D_{ij} between two ellipses is visualized in 6b. Calculating the distance in this way introduces a penalty for choosing a sonication order where two consecutive ellipses overlaps, that is $R_{ij} + R_{ji} > d_{ij}$. Finding the radii of the ellipses in the correct direction requires the knowledge of the parameters of the ellipses. Fitzgerald *et al.* [89] presented a method for finding these parameters by fitting an elliptic curve to a set of unique points. To find these points, the area of intersection between the conical beam and the NF plane needs to be deduced. A detailed description of the method for finding the intersection ellipses is presented in Appendix A.

The calculated distances, either cell-to-cell or ellipse-to-ellipse, are inserted into a *distance matrix*. The matrix has the following structure

$$\mathbf{D} = \begin{pmatrix} 0 & D_{1,2} & \cdots & D_{1,m} \\ D_{2,1} & 0 & \cdots & D_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ D_{m,1} & D_{m,2} & \cdots & 0 \end{pmatrix},$$

where m is the number cells. The rows signifies the starting nodes and the columns the ending nodes of the arc between cells/ellipses i and j. The distance matrix is used as input in the various LPP approximation algorithms.

Three LPP approximation algorithms were implemented. Pseudo code for the algorithms can be found in Appendix D. The first algorithm is a greedy-like algorithm, the farthest neighbour algorithm, presented in Section 6.1.2. This is a simple algorithm, the performance of which relies much on the structure of the problem set, but it is very quick to produce a feasible solution. The second algorithm is a genetic algorithm specially designed for routing problems. The algorithm was first introduced in [90], where a more detailed description of the algorithm can be found. The problem was formulated as a maximization problem, and the fitness value was the length of the path. The strength of specially designed GA is that the operators are closed, meaning that they never produce an infeasible chromosome and no time-consuming repair-heuristics are needed to fix chromosomes that does not fulfil all constraints. To further improve the algorithm, the judgement day and 2-opt local search operators were used. The last algorithm implemented is a BB algorithm as described in Section 6.3.2, using the GLPK Solver to solve the relaxed IP problems.

10 Testing Methodology and Results

A series of tests were performed primarily to evaluate the feasibility of the implemented modules, but also to compare the optimization algorithms and evaluate how they behave with problems of different sizes. Moreover, the treatment planning algorithm was tested in a real clinical case, using manually segmented target volumes and OARs as well as realistic cells, to compare the created plans with a plan produced by an expert.

All work was performed on a Dell Precision T7400 workstation, equipped with an Intel Xenon E5420 2.5 GHz dual-core CPU as well as 4 Gb of RAM-memory and running the Windows 7 64-bit operating system.

10.1 Population Module Algorithms

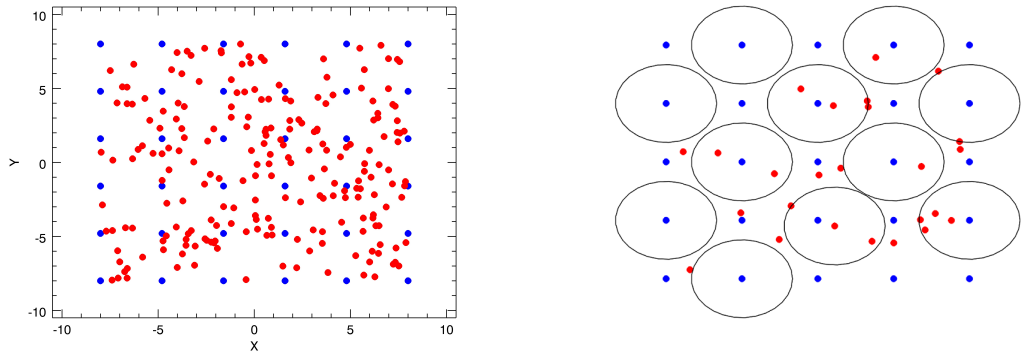
The coverage algorithms were compared in two artificial population cases of different sizes. Both the BUC and the MCUCT problems were examined. Each of the test cases consisted of two sets of cell positions, one ordered and one randomized, distributed in three dimensions inside a cubic volume. Both sets consisted of an equal amount of cells. The cell positions in the larger cases are presented in Figure 7a, projected to the xy-plane. All of the cells used in the test were spherical, with the same radius. For clarity, a simplified example of a BUC population, reduced to 2D, is illustrated in Figure 7b. The target points were distributed uniformly in 3D in a cube one cell diameter larger in every dimension than the cube holding the cell positions. A simple cost estimator was used, which assigned cells with higher x-positions with a linearly higher time cost. The cells with the smallest x-position were given the cost 1 and the rest were given a cost according to their relative x-position. All target points were weighted equally.

The parameters used for constructing the two cases are presented in Table 1. The cases were not designed to resemble a real clinical case and the cost setter has no link to realistic cooling time estimations. For this reason, the interesting result of these tests is not the actual coverage produced by the different algorithms but rather how their computation times compare and vary with problem size and how the algorithms handle 3D population.

Table 1: Description of the population test setup.

Parameter	Large	Small
Cell radius	2.5	1.0
Range for cells positions	$[-8, 8]$	$[-3, 3]$
Number of cell positions	432	250
Range for target points	$[-10.5, 10.5]$	$[-4, 4]$
Number target points	2744	1000
Budget	500	150
Threshold	500	200

The GRASP algorithm was run using a selection parameter $\beta = 0.7$ and five iterations. Two different approaches were taken with the GA. The first version, GA1, did not



(a) (b)

Figure 7: Figure (a) illustrates the cell positions in the large case. All cell positions are projected to the same xy-plane. The blue dots are the ordered set of positions and the red are the randomized positions, all in all 432 positions. Figure (b) shows a simplified population case of similar spatial dimensions as the small case, but reduced to 2D. The illustrated example is a BUC population, limited by the budget constraint.

use the repair function and thus relied only on penalties for constraint handling. The population size and number of generation were chosen to be relatively large. The algorithm was seeded a feasible initial population using a GRASP-like algorithm with $\beta = 0.5$ and only one iteration. The second version, GA2, also penalized infeasible chromosomes, but used heuristic repair every 200th generation to try to keep the chromosomes in the feasible solution space. As the repair operator was computationally quite slow, it was not used on every generation, and the population size was kept small. Moreover, as it was hypothesized that the heuristic repair would be able to guide the algorithm more quickly to a feasible solution space, the number of generations was also kept lower than for GA1. The initialization was random, except for one chromosome which was created using the greedy algorithm. Both versions used roulette wheel selection using SUS and sigma scaling, uniform crossover with a crossover probability of 0.95, and a mutation probability of 0.01. The number of elite chromosomes was 10% of the population size in both cases. The parameters used for the GAs are summarized in Table 2.

Table 2: Parameters for the genetic algorithms.

	GA1	GA2
Population size	100	20
Nr of elites	10	2
Initialization	GRASP, whole population	Greedy, one chromosome
Heuristic repair	No	Every 200th
Iterations	5000	2000

No stopping criterion was used for the GAs. Instead, the algorithms were allowed to evolve through all generations, and the generation in which the best chromosome first was created was recorded and used as a measure of how quickly the algorithm converged.

10.1.1 Results

The results of the BUC and MCUCT tests can be seen in Figures 8 and 9, respectively. In Figure 8 (Figure 9), the normalized coverage (cost), i.e. the coverage (cost) of a subset of cells divided by the coverage (cost) of the best subset of cells for the respective case, of each algorithm is plotted against the computation time. As the computation time required by the BB algorithm in the larger case exceed an hour, its result was omitted. For the GAs, the time required to find the best chromosome for the first time is presented as the computation time. If the GA was not able to improve on the initially seeded population, the total computation time is presented, as was the case for GA2 in both of the MCUCT problems. These numbers gives insight into how quickly the algorithms converged.

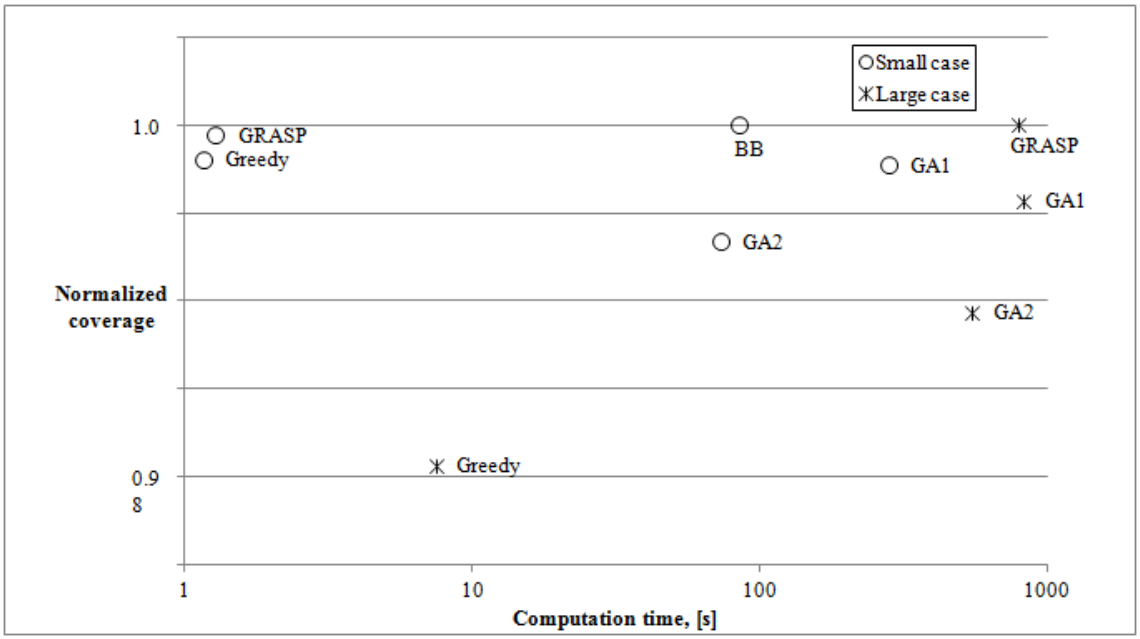


Figure 8: The results of the BUC tests. The normalized coverages, the respective coverage divided by the best coverage in the given case, of the algorithms are plotted against the computation time required, on a logarithmic scale. For the GAs, the time required to find the best chromosome is used instead of the total computation time. The best normalized coverage is 1, and higher is better.

As can be seen from the figures, the greedy algorithm was several orders of magnitude faster than the other algorithms, and was even able to outperform the GAs result wise in some of the cases. As mentioned, GA2 was unable to improve on the result of the initial population in both MCUCT cases. The GRASP algorithm was considerably slower than the greedy algorithm and produced only a slight improvement in the result, more so in the larger case. The bottleneck of the GRASP algorithms computational efficiency is the local

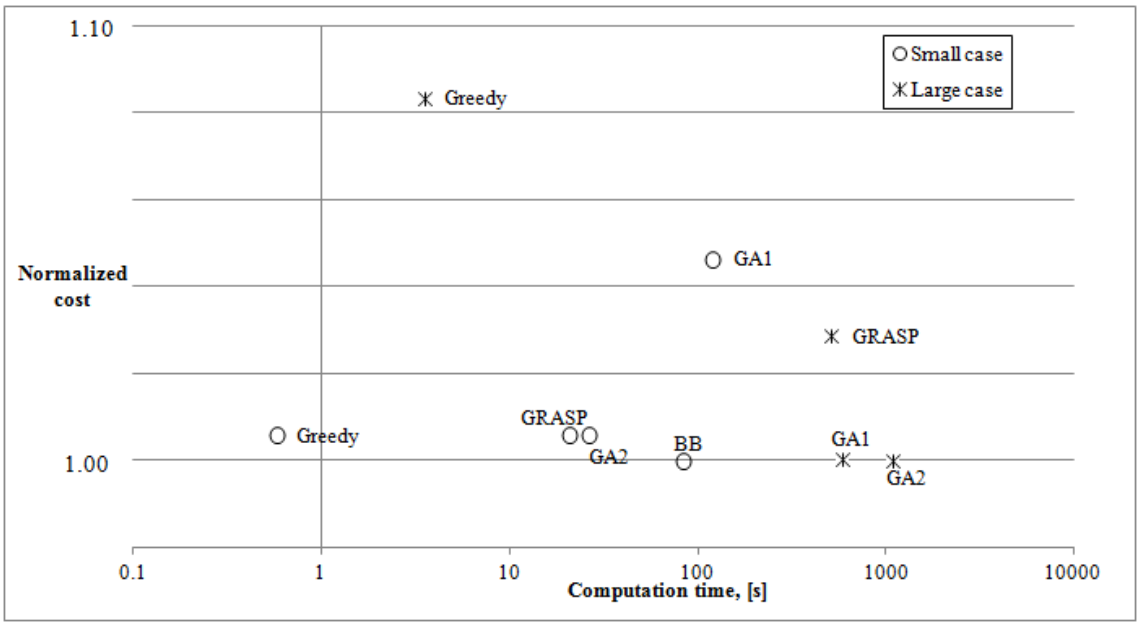


Figure 9: The results of the MCUCT tests. The normalized costs, the respective cost divided by the best cost in the given case, of the algorithms are plotted against the computation time required, on a logarithmic scale. For the GAs, the time required to find the best chromosome is used instead of the total computation time. GA2 was unable to improve on the result of the initial population in both cases. The best normalized cost is 1, and lower is better.

search, which needs to do exponentially larger searches as the problem size increases. This explains the large increase in running time with problem size that can be seen in both the BUC and MCUCT problems.

The GAs inability to perform better than the greedy algorithm, even when utilizing seeding, depicts the problems they had with the overlap, budget, and threshold constraints. The heuristic repair operator is relatively slow, and the small population size limits the effectiveness of the selection and crossover operators. In most cases the GA1 algorithm performed better than GA2, and was close to optimum in the MCUCT case. This raised the question if the repair operator, utilized in GA2, limits the diversity in the population by the way it repairs infeasible chromosomes, by removing and adding cells greedily. This might influence the MCUCT problem more, as cells are first removed and then added until the threshold constraint is satisfied. It should also be noted that GA1 was faster than GA2, even though it had a larger population size and ran for more than double the number of generations. Neither GA1 nor GA2 was particularly sensitive to the size of the problem.

The BB algorithm was able to produce global optimum answers in the smaller cases in reasonable time, but when the problem size grew the computation time became unreasonable. The reason for this is that the number of constraints grows exponentially with the size of the coverage matrix.

The test did simulate a small 3D population case quite well, except that the density of cell positions was somewhat unrealistic. This might have hampered the GAs. The most insightful part of the results is the relative computation times of the algorithms. On the

other hand, as the performance of the algorithms is influenced by several problem parameters (for example the choice of budget and threshold, relative cost and weights *etc.*), it is difficult to draw any conclusion about how they compare performance wise. The test was lacking in that it did not provide much insight about how the parameterized algorithm, namely GRASP and GA, are affected by changes in their respective parameters. Especially GA has several parameters that can be tuned, and finding the optimal combination is not trivial. As the main limitation of the GAs seemed to lie in constraint handling, the used sets of parameters were chosen to emphasize the difference between the use of the heuristic repair operator and penalties.

10.2 Path Maximization Algorithms

The sonication order optimization algorithms were tested in three artificial test cases of different size. The cases consisted of 10, 20, and 30 nodes distributed randomly. The distance matrix was created by calculating the Euclidean distance between the nodes. As a consequence, this test replicates closely the situation in a real clinical case and the results should, therefore, give a good picture of not only how fast the algorithms are, but also how well they perform.

Several versions of the GA were tested: with and without the judgement day and local search operators, as well as with two different population sizes. The JD operator was called by the GA if the best recorded fitness value had not change in 2000 generations, one fourth of the total number of generations. Similarly, the 2-opt local search operator was called if the best fitness had not changed in 400 generations. The population size was equal to the number of nodes, except for one case in which the size was doubled. Again, no stopping criterion was used for the GAs, but they were allowed to evolve until the last generation and the number of generations required to create the best chromosome was recorded. As FN is initialized at a random starting point and as the GA also has a random nature, these algorithms were run in total five times to get an average of their performance. The BB algorithm was only run once.

10.2.1 Results

The results for the sonication order algorithm tests are presented in Figure 10. It is to be noted that the results presented for the FN algorithm and the GAs are averages over five runs. The normalized distance is the distance of a given route divided by the distance of longest (found) route in the given case.

Based on the results presented in the figure, it is easy to identify the characteristic behaviour of the individual algorithms. The NF algorithm, being a greedy-like algorithm, failed to produce results close to the optimum but was several orders of magnitude faster than the other algorithms, with computation times less than milliseconds. The BB, being deterministic, found the optimum for all problems. The algorithm was able to produce results for the smaller cases in reasonable time (1 s and 30 s, respectively), but the computation time exploded to over 10 min in the large case. The GAs performed very well for the smaller problems, regardless of any added heuristics, but only the ones calling the local search operator could produce optimum results for the larger problem. Moreover, the GAs

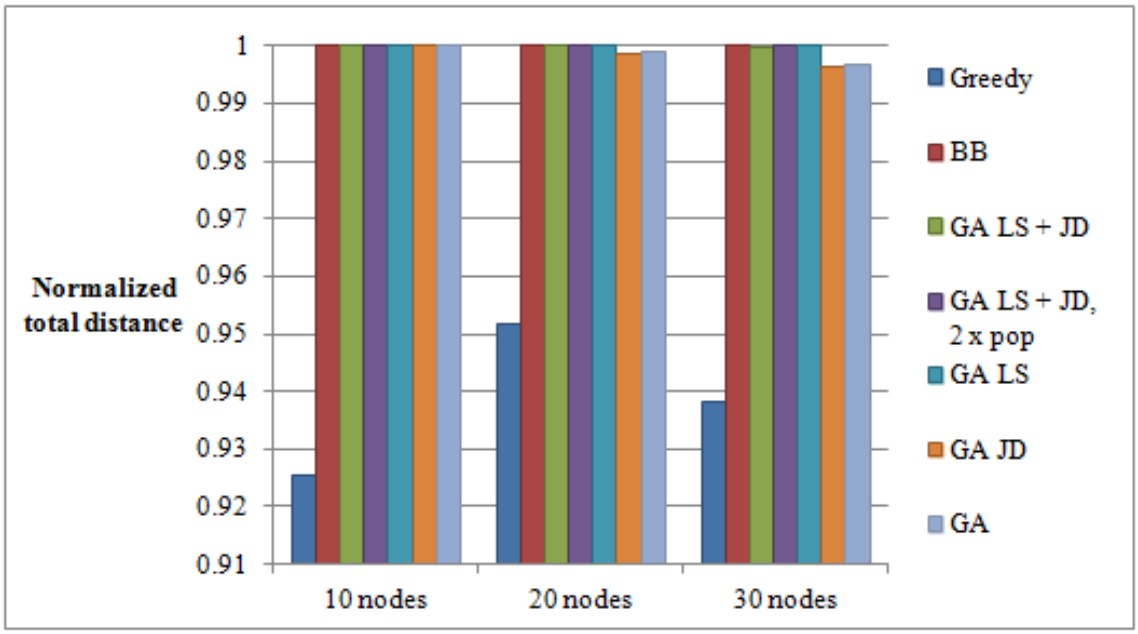


Figure 10: The results of the routing tests. The normalized total distance, the respective total distance divided by the best distance in the given case, of the algorithms are presented for each case. The best normalized total distance is 1, and higher is better. The population sizes in the GAs were the same as the number of nodes, except in one instance in which the size was doubled.

were able to converge very quickly on the large problem especially when compared with the BB algorithm. The times needed to find the best chromosome vary between less than a second (the small case) to almost a minute (the large case). In general, neither the larger population nor the local search operator caused any considerable increase in the computation time. The reason for this is that these algorithms tended to find the best chromosome faster. Based on the results, it can be concluded that the local search operator is necessary for the GA to be successful. The JD operator, on the other hand, improved the performance only marginally. Moreover, it seems that the larger population promotes a good performance.

10.3 Clinical Case

This section presents tests performed based on real clinical planning images. The same images had been used in a real clinical case to plan the delivery of a successful treatment. The primary aim of this test was to determine the feasibility of the initialization and population module prototypes (no sonication order optimization was performed). Secondary aims were to compare the plans generated by the algorithms to the human-made one, as well as to compare the implemented algorithms with each other in a realistic situation. All tests were carried out as BUC problems with a very large budget, which did not constrain the population process. The case was chosen partly due the success of the treatment, implying that the plan produced for the treatment was good, and partly due to the fact that a large majority of the delivered treatments cells were placed in a single cluster. The latter

aspect reduces the number of variables and, thus, simplifies the comparison and allows testing how the algorithms behave when populating a plane.

The fibroid, serosa and OARs in the FF, namely bowels and the spine, were segmented manually based on the planning images. No OARs in the NF were recognised in the images. The segmentation data was then fed to the initialization module, which produces the set of feasible cell positions. The manually segmented fibroid, which was used as the PTV in the tests, was somewhat larger than the PTV defined in the real treatment plan. Furthermore, as the operator had positioned the cells based on expertise and experience, it was not necessary to obey the safety requirement as strictly as the implemented initialization module does. For example, the operator had positioned several cells so that the spine lay inside the FF safety margin. These two factors affect the set of feasible cells that can be delivered and therefore also the outcome of the population procedure. The algorithm used the default safety parameters for the same SW version used in the delivery of the treatment. Last, the module implemented a newer ATA model than the one available in the SW version in question, but as all the cells were positioned in a region which both ATA models contained, this should have no significant effect on the outcome of the plan.

In order to investigate how the algorithms handle problems of different sizes, the tests were first run on three sets with different densities of feasible positions: a small, a medium, and a large problem. As the real case used a single plane for the cells, the cell positions in the sets were also defined in only one plane. The respective separation between the positions for each set is presented in Table 3. As the cluster in the real case was tilted slightly (-5.04^o) about the y-axis (left-right direction), the produced cell sets were also tilted to mimic the case as closely as possible. A simple time cost estimator was used, assigning a linearly higher cost to cells with higher x-coordinates (anterior-posterior direction). A fourth set of cell positions was also created, with 2 mm separation but without tilting, in order to investigate how the algorithms behave when the time cost is uniformly distributed across the cluster. A fifth test, allowing a 1 mm overlap between the cells was also performed. The default cell sizes available in the Sonalleve MR-HIFU system were used (8 mm, 12 mm, and 16 mm diameter cells), so even with the coarsest raster the smallest cells could still overlap each other if placed in adjacent positions. The cells were modelled as ellipses with realistic dimensions. The target points were created 1.25 mm apart on the same plane as the cell positions, so that all were inside the target volume and could potentially be covered by one of the cells. No weighting was used.

Figure 11 presents the set of feasible cells produced by the initialization module for the medium sized tilted case. Due to the nature of the safety checks and parameters, a cell position that is feasible for a given cell size is also feasible for all smaller cell sizes. The problem sizes are presented in Table 3. The number of cell positions and target points vary somewhat, due to the irregular form of the fibroid and the differences in separation between cell positions in the different problems.

The population module was used in two different modes: it either filled the target using all cell sizes simultaneously or using a step-wise procedure, filling the 16 mm cells first and then gradually moving down the scale. Filling all cell size at once allowed the module to optimize the cells better as a whole, but handling more cell position is computationally heavier for the algorithms. The step-wise filling was designed to alleviate the computational strain as much as possible. First, only the 16 mm cell positions and the target points

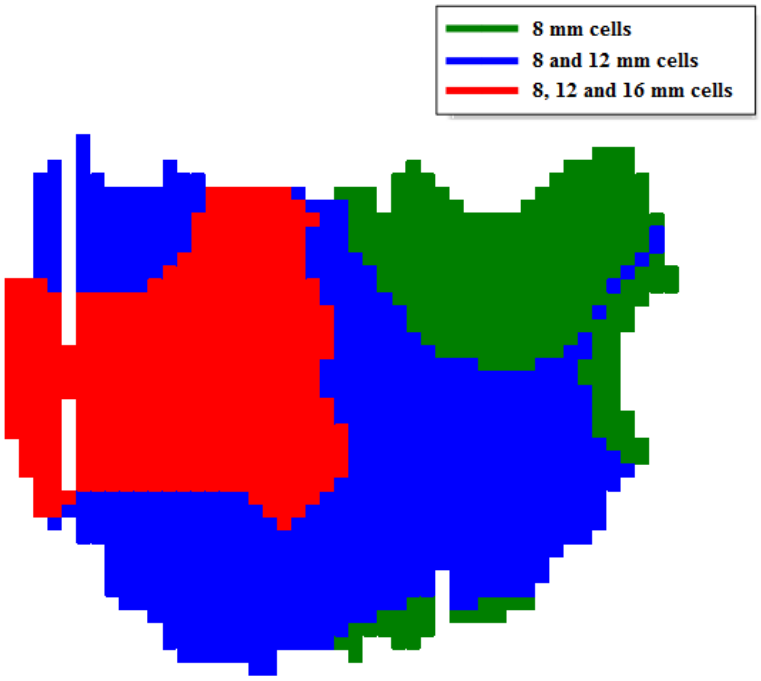


Figure 11: The set of feasible cells in the medium sized non-tilted case visualized in a coronal plane.

Table 3: Description of the clinical test cases.

	Small case	Medium case	Medium case, no tilt	Large case
Position separation	3.5 mm	2 mm	2 mm	1.15 mm
Nr of 16 mm cells	134	415	604	1284
Nr of 12 mm cells	349	1079	1120	3322
Nr of 8 mm cells	444	1376	1357	4200
Total nr of cells	927	2870	3081	8806
Nr of target points	3720	3681	3609	3755

coverable by any on the 16 mm cells were passed to the algorithm. Once the 16 mm cells had been optimized and before the 12 mm cells were passed to the algorithm for optimization, the set of feasible 12 mm cells was updated by removing any of the positions that would cause overlapping with any of the chosen 16 mm cells. This process was repeated after the optimization of the 12 mm cells for the 8 mm cells, removing those positions that would cause overlapping with any of the already chosen 12 mm and 16 mm cells. While the step-wise filling allows splitting the problem into sub-problems that are easier to handle, the solution found by the algorithms in this way is, in general, not the global optimum for the problem. Nevertheless, it was found that in most cases the use of step-wise filling was required for some of the algorithms to be able to produce any solution at all or in a reasonable time.

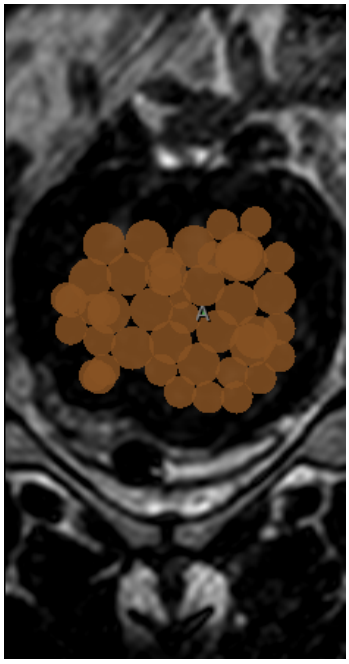
All of the implemented coverage optimization algorithms, described in Section 9.2, were included in the test. The GRASP algorithm was run using 20 iterations and $\beta = 0.7$. Two versions of the genetic algorithm were tested. GA1 used a larger population (50 chromosomes) and more generations (10 000) but no repair function, while GA2 used a smaller population (30 chromosomes) and less generations (6000), but tried to improve on the result by using a heuristic repair operator. The different GAs were chosen to test two different constraint handling techniques in practice and to compare them. Both versions used GRASP seeding ($\beta = 0.5$), a crossover probability of 95 %, a mutation probability of 2 %, sigma scaling and roulette wheel selection with SUS, and ten elite chromosomes.

10.3.1 Produced Plans and Performance of the Algorithms

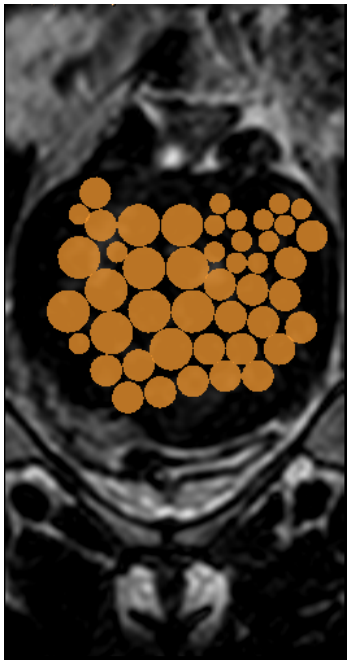
In Figure 12, some of the plans produced are compared to the original plan used to deliver the treatment, shown in Figure 12a. When doing the comparison, it should be noted that the algorithms were applied on a larger PTV than the original. Furthermore, in the real plan the safety requirements were interpreted flexibly, and 16 mm cells were positioned in areas where the algorithm only allows 8 mm cells. Figure 12b visualizes the plan produced by the BB algorithm using step-wise filling and the medium raster, the best plan produced for the given case. A plan produced by the GA1 algorithm using the small raster is presented in Figure 12c. Considering the coarser distribution of feasible cell positions, it is clear that the gaps between the cells tend to be larger than in Figure 12b. The gaps between the cells present in both Figure 12b and 12c were characteristic for all plans in which overlap was prohibited. Figure 12d shows a plan produced using the greedy algorithm and the medium raster as well as by allowing a 1 mm overlap between the cells. This plan resembles the real case more and is able to obtain a far more extensive coverage than the former two cases by filling the gaps between the cells more efficiently.

A comparison of the algorithms is shown in Table 13. The only algorithm that was able to produce results while filling all cells simultaneously was the greedy algorithm. The GAs, on the other hand, failed to improve on the fitness of the initial population, while the GRASP and BB algorithms required an infeasible computation time to produce a solution. For this reason, all results obtained with these algorithms were acquired using step-wise filling. Moreover, both GAs failed to improve on the fitness of the initial population in the large case, despite using step-wise filling, and therefore their results were discarded from the figure.

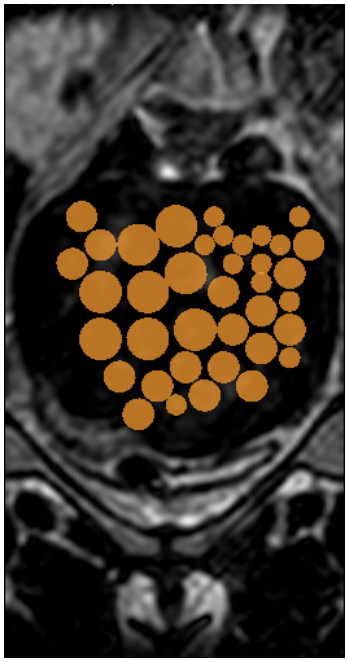
Based on the results presented in Figure 13, some general observations can be made about the performance of the algorithms. First and foremost, allowing a small overlap between the cells leads to a considerable increase in the covered area, as it provides more freedom to position the cells and allows the algorithms to fill the gaps between the cells better. Similarly, using a finer raster allows more accurate positioning of the cells, which consequently leads to slightly better results in the large case. The disadvantage of using a finer raster is, however, the increase in computation time. It can also be seen that filling all cell sizes simultaneously tends to provide better outcome than step-wise filling. Last, the BB algorithm produced the best results in a majority of the cases. However, as the BB algorithm only was able to produce plans using step-wise filling, the produced plans are not global optima, even though the algorithm is deterministic. This fact is demonstrated



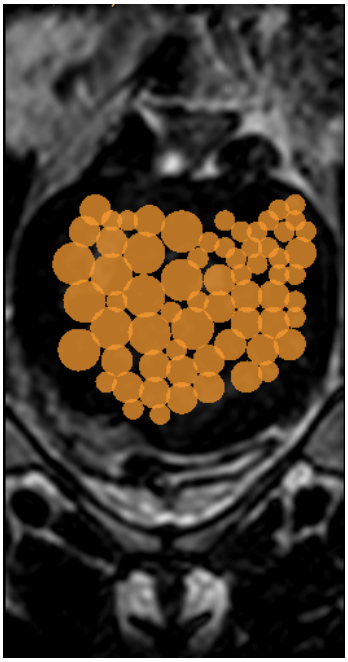
(a) Original plan.



(b) BB algorithm and medium raster.



(c) GA1 and small raster



(d) Greedy algorithm, overlap and medium raster.

Figure 12: Three produced plans compared with the original plan.

in the overlap case, where the GA1 algorithm manages to outperform the BB algorithm, even though both algorithms used step-wise filling.

It can be seen in Figure 13 that, even though some algorithms tended to perform better than others, the plans produced by the algorithms for a given case were difficult to distinguish from one another. Using more complicated algorithms, which require considerably more computation time to produce marginally better plans, might, therefore, not be preferable from a user experience point-of-view. However, without simulated delivery of the plans and further investigation into what a *good enough* plan is, it is difficult to make any far-reaching conclusions in the matter.

As was expected, the relative differences between the running times of the algorithms were similar to those presented in Section 10.1.1. Furthermore, step-wise filling reduced the running time considerably in all cases and for all algorithms, and it was even required in order for some algorithms to produce a solution in reasonable time. The greedy algorithm produced plans, filling all cell sizes simultaneously, in 18 s in the smallest case and 3.6 min for the largest case. Using step-wise filling, the running time dropped by and order of magnitude. The running time of the GRASP algorithm, which directly depends on the number of iterations the algorithm is run, ranged between 30 s to 8 min. Similarly, the time required by the two GAs depends strongly on the choice of algorithm parameters. GA1 required 5–10 min, while GA2 was faster, requiring 2–5 min. The BB algorithm found a solution for the small case in a reasonable 4 min, but required over 50 min to solve the large problem.

The greedy algorithm was highly effective, especially considering its simple structure, and was able to produce as good as or even better plans than the GAs and GRASP al-

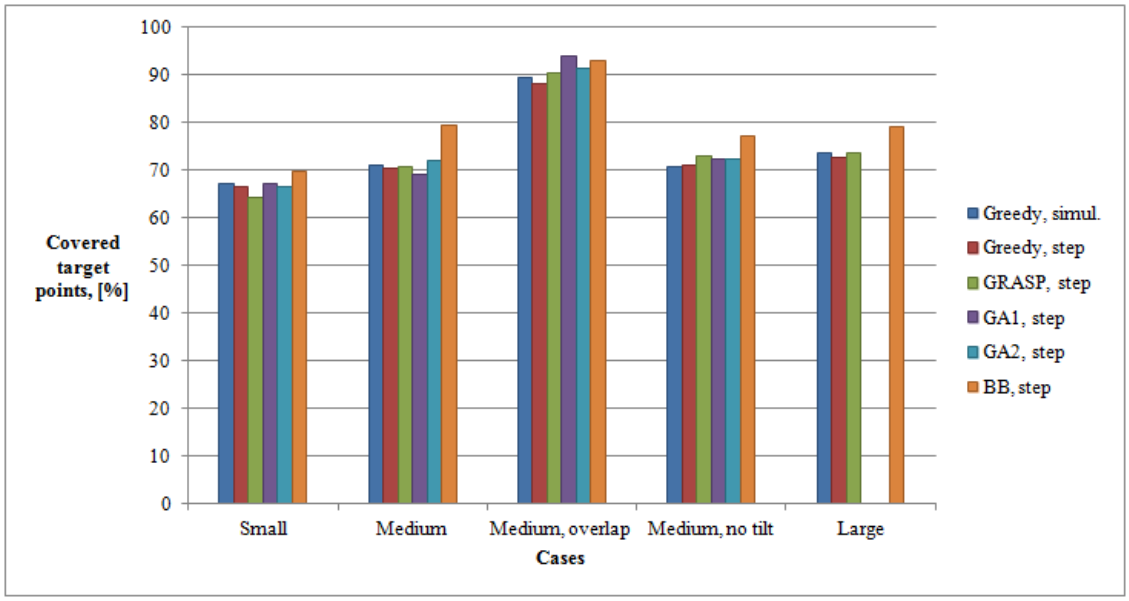


Figure 13: The performance of the algorithm in five different cases. Algorithm names preceded with *step* signifies the algorithm that performed step-wise filling of cells, while *simul.* denotes simultaneous filling of all cell sizes. No results were recorded for the GAs on the large case as they were unable to improve on the fitness of the initial seeded population.

gorithms in most cases. The algorithm also proved to be very fast, using both step-wise and simultaneous filling. The algorithm encountered slight difficulties with the not tilted cluster, in which the cost distribution across the cluster was uniform. A uniform cost and weight distribution implies that many of the feasible cells have equally good weighted coverage-to-cost ratio, which means that the greedy algorithm selects a cell at random. As a consequence, the algorithm fails to pack the cells tightly, and the performance suffers. The `GreedyBUC_maxcov` heuristic encounters similar problems even in the tilted cases, in which the cost distribution is not uniform. This renders the value it adds to the greedy BUC algorithm questionable.

Even though the construction phase of the GRASP algorithm has a similar approach to cell selecting as the greedy algorithm, the local search seems to allow the algorithm to correct most of the loose packing in the not tilted case and consequently produce a good result. Based on results, it can be seen that the algorithm can improve slightly on the result of the greedy algorithm, but on the other hand requires considerably longer time to produce the solution. As the number of target points and selected cells remained relatively constant between the cases, the running time of the algorithm increases almost linearly with the number of cell positions to choose from. It was found, however, that this caused simultaneous filling impractical when using the medium and large rasters.

The BB algorithm was able to produce the best plans for most cases, which was to be expected from a deterministic algorithm. As the algorithm only produced results using step-wise filling, the results found were not, however, the global optima. On the other hand, the algorithm was also the most time demanding, with the running time exceeding 50 min for the large case. It can, therefore, be concluded that the algorithm, in its current form, is too sensitive to the problem size. It should be noted that at least in the smaller cases a majority of the computation time is used in a preparatory stage which constructs the problem based on a MathProg model, while the problem solving was relatively quick. It would therefore be interesting to investigate how the performance of the algorithm changes if the GLPK library was called directly, without using the GLPK Solver.

The robustness of the GA algorithm suffers from the difficulties with the overlap constraints. This is implied by the algorithms inability to improve on the fitness of the initially seeded population on the larger problems, even when using step-wise filling. On the other hand, when the raster is coarse enough, as in the step-wise filling of the small case, the algorithm improved the results in all three steps. A similar effect can be seen in the overlap case, in which the cells can be placed more freely and the probability of not satisfying the constraints decreases. To be able to improve on the results, the algorithm requires either the use of a repair operator or that the solution space is sampled through many generations. Both approaches tend to prolong the computation times. Based on the results, it is difficult to form any preferences about which constraint handling method to use, as both GAs produced results that were at par with the greedy algorithm.

11 Discussion

The most important result of the tests described in Section 10, is that all the modules incorporated in the prototype algorithm proved their feasibility in tests that mimic real clinical cases. This section reflects on the results in further detail. Also, a number of improvements for the algorithms as well as some further investigations are proposed.

Greedy algorithms The greedy(-like) algorithms showed their strengths in both the coverage and routing problems by constructing feasible solutions orders of magnitude faster than the other algorithms. Especially in the 3D coverage problem and the clinical case, the algorithm was able to produce results that were as good as or even better than the GAs. The result of the FN algorithm was, however, not as good in the LPP test.

It would seem that the inherent symmetry in the coverage problems favours the greedy algorithm, and based on the good results and superior running time the algorithm showed good potential for being used as BUC and MCUCT problem solvers. In the non-tilted clinical case an apparent weakness of the greedy algorithm presented itself. As the range of cell sizes and forms was very limited and as the distribution of costs and weights over the plane was uniform, many cells were equally good choices. As a consequence, the greedy algorithm selected many cells at random instead of packing them tightly. The algorithm did not encounter similar problems in the 3D case or the clinical cases with tilted clusters, as the cost of the cells varied linearly as a function of their x-position. This allowed the algorithm to prioritise the cheaper cells and start packing from the one edge(/face) of the cluster(/cube). The `GreedyBUC_maxcov` heuristic, which neglects cost distributions altogether, is even more inclined to random positioning, as it does not consider varying costs. For this reason, it contributes very little the overall performance of the `GreedyBUC` as long as no weighting is used to prioritize a certain part of the target.

In order to guide the greedy heuristics towards minimizing the gaps between the cells, be the cost distribution uniform or not, tight packing needs to be rewarded. This could either be implemented as a factor in a multi-objective ordering criterion or as a secondary criterion, to distinguish between cells with equally good ratios. One way of implementing such a criterion would be to, for example, compare the positions of the candidate cells to the (weighted) mean position of the chosen cells and select the closest one. Another way, which would remove the need for any further ordering criterion, would be to introduce weights on the positions close to cells that have been chosen. In the MCUCT case, the problem might, however, be different as it is not clear whether the chosen cells should preferably be packed tightly or spread out for the treatment to be most effective.

The FN algorithm did not perform as well in the LPP as the greedy algorithm did in the coverage problems. The reason for this is that the greedy addition of nodes to the route tends to cause the last nodes that are added the route to be relatively short, which consequently harms the final result. The extent of this deteriorating effect is, however, dependent on the structure of the problem and the relative positions of the nodes to each other. Nevertheless, there are no obvious solutions for this problem that does not involve changing the nature of the algorithm. Iterating through all nodes, using them all as starting nodes, and selecting the best route from the ones produced could bring a marginal improvement, but the problem would still remain. Testing the greedy GR algorithm, described in Section

6.1.2, would also be interesting. The algorithm is, none the less, still a good constructor of feasible solutions and could e.g. be used to create seeding populations for the genetic algorithm.

GRASP algorithms Based on the result, larger problem sizes and more varied cost distributions seems to favour the GRASP algorithm in the coverage problems. The construction phase of the GRASP algorithm is prone to encounter similar problems as the greedy algorithm with uniform cost distributions, and allowing the algorithm to choose from an even wider range of cells essentially renders the problem worse. This disadvantage is somewhat compensated for by the local search procedure. Nevertheless, the construction phase of the algorithm should be improved to encourage tight packing in a similar way as the greedy algorithm.

The local search procedure for the BUC problem has a weakness in that it only swaps cells and does not attempt to add any cells to the original solution. This does not promote coverage maximization; it only reduces the cost. Instead, the procedure should attempt to pack the cells as tightly as possible so that more cells can be added to the solution. Therefore, the logic of the local search procedure should be re-designed. The current local search procedure for the MCUCT problem serves it purpose better.

Genetic algorithms In the coverage problem tests, both of the genetic algorithms struggled with improving on the fitness of the initially seeded population. The most obvious reason for this is the problematic constraints handling required by the overlapping and budget/threshold constraints. In the 3D case, the GA1 algorithm, not utilizing the heuristic repair operator, seemed to perform better as it was given a large population size and an adequate number of generations to evolve. The poor result of GA2, utilizing the repair operator, especially in the MCUCT case, is believed to originate in the way the repair operator produces feasible solution from infeasible ones. The repair function designed for the MCUCT problem first removes cells until no overlap exists, after which it adds new cells greedily until the threshold constraint is satisfied. If the procedure removes a majority of the cells from the original chromosome, almost all of the remaining cells are available for re-selection. The greedy choice of cells will, therefore, often select the same cells to be added to the chromosome. Consequently, the repair function reduces diversity in the population, which in turn effectively limits the direction in which the algorithm is able to develop. In the non-tilted clinical case, in which the repair operator adds cells to the population almost randomly (as the cost distribution is uniform), the GA is able to produce better results than the greedy algorithm. It would thus seem that the operator does not restrict the diversity which in turn favours the evolution of the population. The GAs perform better in the clinical cluster case than in the 3D case, as the probability of not satisfying the overlap constraint decreases when the cell positions are distributed over a larger plane than when packed into a small cube. However, it still struggled with the finer rasters – the closer the cells positions are to each other, the more probable overlapping becomes. In practice the GAs required the use of step-wise filling in order to improve on the initial fitness of the population. Another problem with the heuristic repair is the computation time it requires, which in practice makes it impossible to use after the creation of every generation.

The best alternative to improve the performance of the GAs would be to, if possible, design closed operators that never produce infeasible solutions. If that fails, the constraint handling needs to be improved. Introducing penalties that depend on the extent the violation is one alternative improvement. Moreover, the repair operator should be improved so that it preserves the diversity in the population better. This could possibly be achieved by shifting the focus of the operator away from the cells with poor objective values and concentrate more on the cells that overlap. By examining the cells that overlap and exchanging the one with worse objective value into a cell that does not cause overlap, the number of cells that need to be removed would hopefully decrease. This would reduce the risk of adding the same cells to all chromosomes in the MCUCT case, and possibly also improve the running time of the algorithm. Implementing a GRASP-like addition and removal of cells (pursuing one of the good alternatives, and not necessarily the best one) could help to introduce more diversity into the selection process. The computational efficiency of the procedure also needs to be improved, so that it can be utilized more often and with larger population.

Further investigation should also be done into how the numerous parameters and different operator types influence the performance of the GAs. Only two different GAs were applied in the test, with an emphasis on evaluating different constraint handling techniques, which did not allow good insight into how the other controlling parameters should optimally be chosen. Parameter optimization for GAs is a complicated task [44, 77], but an attempt should be made to find an adequate set of parameters for all instances of the coverage problems.

In contrast, the GA used for solving the LPP proved very efficient, producing close-to-optimal results in reasonable time even for larger problems. The results improved considerably when the 2-opt local search operator was used. The only parameters that the algorithm needs are the population size as well as the threshold for when to initiate the local search and judgement day operators, which simplifies the parameter optimization considerably. Implementing better local search algorithms, such as 3-opt local search, and improving their run time would presumably further improve the performance.

Branch-and-bound algorithms The BB algorithm failed to produce results in reasonable time both for the 3D coverage problem and the larger routing problem. It was, however, able to produce the best plans in most of the clinical cases, albeit only using step-wise filling. Moreover, computation time grew quickly with the problem size. For this reason it can, in its current form, neither be recommended for population nor sonication order optimization purposes. As the time mostly goes into loading the MathProg models, it would be interesting to test its performance by calling the GLPK library directly, without utilizing the GLPK Solver. Also, reformulating the algorithm, where possible, into a branch-and-cut algorithm could also improve the running time.

General The goodness of a plan is determined by the quality of the treatment it defines. Therefore, it is difficult to draw objective conclusions based on the plans produced by the algorithms in the clinical case, without conducting extensive simulations or clinical tests. Most of the plans appeared quite similar in visual inspection, even though their coverage varied somewhat. Therefore, it would be beneficial to investigate what a *good enough* plan

is, and how the small variations in the coverage between the plans influence the quality of the treatment. More specifically, it would be interesting to compare the quality of the treatments based on a plan produced by a greedy algorithm and a GA. The GA requires several orders of magnitude more time to produce a plan that is only slightly better than the plan produced by the greedy algorithm. It is also unclear whether the gaps between the cells should be minimized or distributed evenly, and whether e.g. the centre of the target should be prioritized. Even though this knowledge would exist among the physician delivering the treatments, very little published research is available on the matter. This makes it challenging to integrate the expertise into a mathematical model for the initialization and the population module. At the moment, more research and knowledge about optimal cell positioning is required to eventually develop the population algorithm further.

Even though most of the produced plans resembled each other, all of them differed from the original plan used to deliver the treatment. The reason for this was partly the smaller PTV used in the original plan, but the operators flexible interpretation of the safety limits probably also contributed. The operator evaluated the situation, using his/her expertise and experience, and determined that it was safe to place cells closer to OARs in the FF than the SW safety margin had allowed. This serves to prove that an algorithm cannot replace the judgement of an experienced operator.

It is also difficult to comment on the absolute speed of the algorithms given that IDL is neither optimized for speed nor looping. Implementing the algorithm, and especially bottleneck procedures, in a programming language better optimized for loops would improve on their performance considerably. Step-wise filling of cells simplified the filling process and improved the computation time for all algorithms, but as it does not, in general, tend to the global optimum the simultaneous cell filling should be preferred. Last, it was observed that finer rasters allow more accurate cell positioning and consequently better coverage but also require more computation time. The raster accuracy should therefore be balanced to find the point of diminishing returns.

To conclude, the greedy algorithm is recommended for solving coverage problems. Several suggestions have, however, been given to improve the performance of all algorithms, and the final choice of algorithms depends on how they can be improved performance and efficiency wise. The branch-and-bound algorithm was able to produce optimal solutions in both the coverage and the routing problems, but was in its current form very sensitive to the problem size. Calling the GLPK library directly or formulating the algorithm into a branch-and-cut algorithm might improve on the performance enough to be practical. The heuristics used in the greedy and the GRASP algorithms for coverage problems should be modified to better manage situations in which the weight and cost distribution is uniform and should reward tight packing more. Also, the logic of the local search procedure utilized in the GRASP algorithm for BUC problems needs to be revised. The GAs difficulties with constraint handling needs to be resolved, either by introducing more adaptable penalties or improving the repair operator. It would also be interesting to test simulated annealing, which has proven to be as good as or even better than GAs at solving some problems.

Based on the results of the routing tests, it is recommended that the sonication order optimization is done using the implemented GA, either directly or with some small enhancements.

Part IV
Summary

The aim of this thesis was to design an outline for an automatic treatment planning algorithm for Magnetic Resonance guided High Intensity Focused Ultrasound (MR-HIFU), and to produce a prototype of such an algorithm, evaluating and comparing alternative methods of implementation. The work was carried out by studying similar modalities, namely high dose rate (HDR) brachytherapy and network optimization, and applying the gained understanding to MR-HIFU. The presented algorithm outline is general enough to be applied to any MR-HIFU ablation application. The implemented algorithm was tested in two artificial cases and one real clinical case to evaluate its feasibility, to compare the alternatives, and to locate areas of further development.

The presented outline for a treatment planning algorithm is based on the same principal structure as HDR planning algorithms: acquiring planning images, segmenting the target volume and relevant sensitive organs, and optimally filling the target volume with treatment cells. However, due to the complex characteristics of MR-HIFU treatments, the optimization procedure is divided into three separate modules. The first module, the initialization module, creates a set of feasible positions for the treatment cells, while taking into consideration all safety issues and physical limitations associated with delivering a MR-HIFU treatment. The next module, the population module, fills the target volume optimally by selecting from the set of feasible cell positions produced by the previous module. The module can be asked to either maximize the area/volume covered by the treatment cells, given an upper time limit, or alternatively to minimize the time used to treat a specified part of the tumour. The last module, the sonication parameter module, optimizes the remaining parameters, such as the sonication order, which influence the outcome of the treatment. An important feature of the presented planning algorithm, which the HDR equivalences lack, is the ability to update the plan based on feedback received as the treatment is delivered. Furthermore, the outline is based on an intuitive flow of information, which fits well into the current software framework, and which the users finds easy to learn and adapt to. The presented treatment planning algorithm is, to the writer's knowledge, the first treatment planning algorithm for MR-HIFU that optimizes the treatment and has the ability to update the plan based on feedback.

The implemented prototype algorithm has the ability to produce off-line treatment plans, optimizing the treatment for maximum coverage or minimum time, as well as the sonication order. It also takes into consideration all relevant safety issues and apparatus limitations. Utilizing cumulative heating in the tissue to improve the efficiency of HIFU treatments has recently spurred interest, but very little published research is available on using it in treatment optimization. For this reason, optimization utilizing cumulative heating was not included in the prototype algorithm. Four alternative population algorithms were implemented for comparative purposes: a greedy, a GRASP, a genetic, and a branch-and-bound algorithm. Furthermore, three alternative algorithms were implemented for the sonication order optimization problem: a greedy-like, a genetic, and a branch-and-bound algorithm.

The population algorithms were tested in two cases: an artificial case, in which a three

dimensional volume was filled with spherical cells, and a real clinical case, in which realistic cells were positioned on a single plane inside a segmented tumour. The prototype algorithm was able to produce realistic plans in the clinical case, but their quality is difficult to compare objectively without simulating the treatments. The sonication order algorithms were also tested in a number of artificial cases. The tests gave an indication of how the implemented algorithms compare, but little can be said about their absolute effectiveness before they have been translated to a more suitable programming language. The most important result of the tests, however, was that the whole prototype treatment planning algorithm proved its feasibility.

References

[1] P. Sasieni et al. What is the lifetime risk of developing cancer?: the effect of adjusting for multiple primaries. *British Journal of Cancer*, 105:406–465, 2011.

[2] A. Jemal et al. Global cancer statistics. *A Cancer Journal for Clinicians*, 61:69–90, 2011.

[3] B. Abdullah et al. Magnetic resonace-guided focused ultrasound surgery (MRgFUS) treatment of uterine fibroids. Published online (Cited: November 2011), 2010. URL http://www.biij.org/2010/2/e15.

[4] P. Haigron et al. Image-guided therapy: Evolution and breakthrough. *IEEE Eng Med Biol Mag*, 29:100–104, 2010.

[5] E. Martin et al. High-intensity focused ultrasound for noninvasive functional neurosurgery. *Annals of Neurology*, 66:858–861, 2009.

[6] R. Fedewa et al. Automated treatment planning for prostate cancer HIFU therapy. In *Ultrasonics Symposium*, volume 4, pages 1135–1138, September 2005.

[7] M. Köhler. *Sonication Methods and Motion Compensation for Magnetic Resonance Guided High-Intensity Focused Ultrasound*. PhD thesis, Helsinki University of Technology, Faculty of Information and Natural Sciences, Department of Biomedical Engineering and Computational Science, 2009.

[8] G. ter Haar. High intensity focused ultrasound for the treatment of tumors. *Echocardiography*, 18:317–322, 2001.

[9] F. Duck. *Physical Properties of Tissue*. Academic Press, London, Great Britain, 1990.

[10] I. Dragonou et al. Non-invasive determination of tissue thermal parameter from high intensity focused ultrasound treatment by volumteric MRI thermometry. *NMR in Biomedicine*, 22:843–851, 2009.

[11] X. Fan and K. Hynynen. Ultrasound surgery using multiple sonications - treatment time considerations. *Ultrasound in Medicine & Biology*, 22:471–482, 1996.

[12] C. Ho et al. Thermal therapy for breast tumors by using a cylindrical ultrasound phased array with multifocus pattern scanning: a preliminary numerical study. *Physics in Medicine and Biology*, 52:4585–4599, 2007.

[13] J. Enholm et al. Improved volumetric MR-HIFU ablation by robust binary feedback control. *IEEE Transactions on Biomedical Engineering*, 57:103–113, 2010.

[14] D. Li et al. A study of heating duration and scanning path in focused ultrasound surgery. *Journal of Medical Systems*, pages 1–8, 2010. URL http://dx.doi.org/10.1007/s10916-010-9463-6.

[15] C. Mougenot et al. Quantification of near-field heating during volumetric MR-HIFU ablation. *Medical Physics*, 38:272–282, 2011.

[16] C. Mougenot et al. MR monitoring of the near-field HIFU heating. In *American Institute of Physics Conference Series*, volume 1113, pages 159–161, April 2009.

[17] S. Dromi and oth. Pulsed-high intensity focused ultrasound and low temperature-sensitve liposomes for enhanced targeted drug delivery and antitumor effect. *Clinical Cancer Research*, 13:2711–2727, 2007.

[18] F. Curra and L. Crum. Therapeutic ultrasound: Surgery and drug delivery. *Acoustical Science and Technology*, 24:343–348, 2003.

[19] W. Pitt et al. Ultrasonic drug delivery - a general review. *Expert Opinion on Drug Delivery*, 1:37–56, 2004.

[20] K. Hynynen. Ultrasound for drug and gene delivery to the brain. *Advanced Drug Deliver Reviews*, 60:1209–1217, 2008.

[21] T. Freeman. Vessel ablation enhances uterine fibroid ablation. Published online (Cited: November 2011), December 2010. URL http://www.auntminnie.com/index.asp?sec=ser&sub=def&pag=dis&ItemID=93499.

[22] F. Wu et al. Tumor vessel destruction resulting from high-intensity focused ultrasound in patients with solid malignancies. *Ultrasound in Medicine & Biology*, 28: 535–542, 2002.

[23] J. Xiang et al. Unequal heating duration to reduce the treatment time in high-intensity focued ultrasound therapy. In *IEEE International Ultrasonics Symposium Proceedings*, 2009.

[24] A. Payne. *Minimization of Themral Dose Delivery Time and Development of an Isolated Kideny Phantom: Application for High Intensity Focused Ultrasound*. PhD thesis, University of Utah, Department of Mechanical Engineering, 2008.

[25] H. Liu et al. A fast and conformal heating scheme for producing large thermal lesions using a 2D ultrasound phased array. *International Journal of Hyperthermia*, 23:69–82, 2007.

[26] C. Mougenot et al. MR-HIFU enhanced volumetric ablations. In *Proceedings of the International Symposium on Theraputic Ultrasound*, 2010.

[27] L. Hui et al. Treatment planning of scannig time and path for phased high-intensity focused ultrasound surgery. In *2nd International Conference on Biomedical Engineering and Informatics, 2009. BMEI '09*, volume 2009, pages 1–4, 2009.

[28] A. Blankespoor. *Model Predictive Control of Focused Ultrasound Treatment in Three Dimensions for Treatment Time Reduction*. PhD thesis, University of Utah, Department of Mechanical Engineering, 2008.

[29] M. Malinen et al. Scannig path optimization for ultrasound surgery. *Physics in Medicine and Biology*, 50:3473–3490, 2005.

[30] Philips Medical Systems. MR-HIFU safety training material, November 2009.

[31] P. White et al. A pre-treatment planning strategy for high-intensity focused ultrasound (HIFU) treatments. In *Ultrasonics Symposium*, pages 2056–2058, 2008.

[32] F. Galea and C. Roucariol. Mathematical modelling of HDR/PDR brachytherapy treatment planning problems. Technical report, Université de Versailles, 2004.

[33] R. Meyer and other. MIP models and BB strategies in brachytherapy treatment optimization. *Journal of Global Optimization*, 25:23–42, 2003.

[34] I. Kolkman-Deurloo et al. Anatomy based inverse planning in HDR prostate brachytherapy. *Radiotherapy & Oncology*, 73:73–77, 2004.

[35] E. Lessard and J. Pouliot. Inverse plannig anatomy-based dose optimization for HDR-brachytherapy of the prostate using fast simulted annealing algorithm and dedicated objective function. *Medical Physics*, 28:773–779, 2001.

[36] N. Jain and M. Kahn. Clinical decision-support systems in radiation therapy. In *First International Symposium on 3D Radiation Treatment Planning*, 1993.

[37] S. Miller et al. Brachytherapy cancer treatment optimization using simulated annealing and artificial neural networks. In *Canadian Conference on Electrical and Computer Engineering*, 2001.

[38] M. Hosseini-Ashrafi et al. Pre-optimization of radiotherpay treatment planning: an artificial neural network classification aided technique. *Physics in Medicine and Biology*, 44:1513–1528, 1999.

[39] S. Petrovic et al. A novel case based reasoning apprach to radiotherapy planning. *Expert Systems with Applications*, 38:10759–10769, 2011.

[40] H Ruotsalainen et al. Interactive multiobjective optimization for anatomy-based three-dimensional HDR brachytherapy. *Physics in M*, 55:4703–4719, 2010.

[41] E. Amaldi. Optimizing base station siting in UMTS networks. In *Proceedings of IEEE Vehicular Technology Conference*, 2001.

[42] E. Agustín-Blas et al. A hybrid grouping genetic algorithm for citywide ubiquitous WiFi access deployment. In *IEEE Congress on Evolutionary Computation*, 2009.

[43] N. Weicker et al. Evolutionary multiobjective optimization for base satation transmitter placement with frequency assigment. *IEEE T Evolut Comput*, 7:189–203, 2003.

[44] J. Munyaneza et al. Optimization of antenna placement in 3G networks using genetic algorithms. In *Third International Conferecne on Broadband Communiations, Information Technology & Biomedical Applications*, 2008.

[45] H. Meunier et al. A multiobjective genetic algorithm for radio network optimization. In *Proceeding of the 2000 Congress on Evolutionary Computation*, 2000.

[46] S. Hurley. Planning effective cellular mobile networks. *IEEE Transactions on Vehicular Technolo*, 51:243–253, 2002.

[47] C. Lee and H. Kang. Cell planning with capacity expansion in mobile communications: a tabu search approach. *IEEE T Veh Technol*, 49:1678–1691, 2000.

[48] P. Calégari et al. Parallel island-based genetic algorithm for radio network design. *Journal of Parallel Distributed Computing*, 47:86–90, 1997.

[49] D. Amzallag et al. Capacitated cell planning of 4G cellular networks. Technical report, Technion Computer Science Department, 2007.

[50] C. Ting et al. Wireless heterogeneous transmitter placement using multiobjective varibale-length genetic algorithm. *IEEE Transactions on Systems Man and Cybernetics Part B*, 39:945–958, 2009.

[51] K. Lieska et al. Radio coverage optimization with genetic algorithms. In *The Ninth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 1998.

[52] C. Hu et al. Optimal deployment of distributed passive measurement monitors. In *IEEE International Conference on Communications*, 2006.

[53] K. Suh et al. Locating network monitors: Complexity heuristics and coverage. In *Proceedings of IEEE Infocom*, 2005.

[54] K. Chakrabarty et al. Grid coverage for surveillance and target location in distributed sensor networks. *IEEE T Comput*, 51:1448–1453, 2002.

[55] S. Habib. Modeling and simulating coverage in sensor networks. *Computer Communications*, 30:1029–1035, 2007.

[56] S. Ali. An efficient branch and cut algorithm for minimization of number of base stations in mobile radio networks. In *The 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communicatons*, 2002.

[57] F. Neumann and C. Witt. *Bioinspired Computation in Combinatoral Optimization*. Springer-Verlag, Berlin, Germany, 2010.

[58] H. Moser et al. The parameterized complexity of the unique coverage problem. In *Proceedings of the 18th International Symposium on Algorithms and Computation*, 2007.

[59] S. Khuller et al. The budgeted maximum coverage problem. *Information Processing Letters*, 70:39–45, 1999.

[60] W. Fernandez de la Vega and M. Karpinski. On the approximation hardness of dense TSP and other path problems. *Information Processing Letters*, 70:53–55, 1999.

[61] D. Amzallag. Coping with interference: From maximum coverage to planning cellular networks. In *Proceeding of the 4th Workshop on Approximation and Online Algorithms*, 2006.

[62] D. Curtis et al. Budgeted maximum coverage with overlapping costs: Monitoring the emerging infectious network. In *ALENEX*, 2010.

[63] J. Bleiholder et al. Query planning in the presence of overlapping sources. In *Proceedings of the 10th International Conference on Extending Database Technology*, 2006.

[64] D. Demaine et al. Combination can be hard: Approximability of the unique coverage problem. In *Proceeding of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2006.

[65] G. Laporte. The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59:231–247, 1992.

[66] Q. Wu et al. The NPO-completeness of the longest hamiltonian cycle problem. *Information Processing Letters*, 65:119–123, 1998.

[67] R. Torres-Velázquez and V. Estivill-Castro. Local search for hamiltonian path with applications to clustering visitation paths. *Journal of the Operational Research Society*, 55:737–748, 2004.

[68] T. Rintala. Empirical comparison of stochastic algorithms. Published online (Cited: November 2011), Jul 1996. URL http://lipas.uwasa.fi/cs/publications/2NWGA/node38.html.

[69] T. Cormen. *Introduction to Algorithms*. The Massachusets Institute of Technology, Cambridge, MA, USA, 2nd edition, 2001.

[70] M. Resende. Computing approximate solutions of the maximum covering problem with GRASP. *Journal of Heuristics*, 4:161–171, 1998.

[71] P. Festa and M. Resande. *Wiley Encyclopedia of Operations Research and Management Science*, chapter "Effective application of GRASP". John Wiley & Sons, Inc., 2010.

[72] D. Johnson. Local optimization and the traveling salesman problem. In *Proceedings of the Seventeenth International Colloqium on Automata, Languages and Programing*, 1990.

[73] D. Johnson. Experimental analysis of heuristics for the STSP. In *Local Search in Combinatorial Optimization*, 2001.

[74] G. Gutin and D. Karapetyan. Greedy like algorithms for the traveling salesman problem and multidimnesional assignment problems. Published online (Cited: November 2011), March 2009. URL http://eprints.pascal-network.org/archive/00005211/01/GreedyChapter3.pdf.

[75] G. Gutin et al. Traveling salesman should not be greedy: Domination analysis of greedy-type heuristics for the TSP. *Discrete Applied Mathematics*, 117:81–86, 2001.

[76] J. Bang-Jensen et al. When the greedy algorithm fails. *Discrete Optimization*, 1: 121–127, 2004.

[77] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998.

[78] R. Haupt and S. Haupt. *Practical Genetic Algorithms*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2004.

[79] P. Larrañaga et al. Genetic algorithms for the traveling salesman problem: A review of representation and operators. *Artificial Intelligence Review*, 13:129–170, 1999.

[80] Z. Michalewicz. Genetic algorithms, numerical optimization, and constraints. In *Proceeding of the 6th International Conference on Genetic Algorithms*, 1995.

[81] S. Louis and G. Li. Augementing genetic algorithms with memory to solve traveling salesman problems. In *Proceedings of the Joint Conference on Information Sciences*, 1997.

[82] A. Misevicius et al. Improving local search for the traveling salesman problem. *Information Technology and Control*, 36:187–195, 2007.

[83] V. Kureichick et al. Genetic algorithm for solution of the traveling salesman problem with new features against premature convergence. Technical report, Taganrog State University of Radio-Engineering, 1996.

[84] C. Floudas and P. Pardalos. *Encyclopedia of Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[85] B. Liu et al. Triangular mesh model reconstruction from scan point clouds based on templates. *Tsingua Science and Technology*, 14:56–61, 2009.

[86] D. Cook. *The Theory of the Electromagnetic Field*. Courier Dover Publications, Mineola, N.Y., USA, 2002.

[87] A. van Oosterom and J. Strackee. The solid angle of a plane triangle. *IEEE Transactions on Biomedical Engineering*, 30:125–126, 1983.

[88] D. Eberly. Intersection of a triangle and a cone. Published online (Cited: November 2011), March 2008. URL http://www.geometrictools.com/Documentation/IntersectionTriangleCone.pdf.

[89] A. Fitzgibbon et al. Direct least square fitting of ellipse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:476–480, 1999.

[90] L. Yang and D. Stacey. Solving the traveling salesman problem using the enhanced genetic algorithm. In *Canadian Conference on AI'01*, 2001.

Appendices

Appendix A Plane-cone Intersection

If the angle between the normal of a plane and the axis of a cone is less than the opening angle of the cone, the intersection between two is an ellipse. Below is presented a procedure for finding the intersection between a plane and a cone and estimating the ellipse parameters; the centre point as well as the length and direction of the major and minor axes. The procedure was implemented in the sonication order module to determine the radius of the ellipses in the direction of all other ellipses for calculating the distance matrix.

A plane is defined as

$$\bar{X}(s,r) = \bar{P}_0 + s\bar{E}_0 + r\bar{E}_1, \tag{A.1}$$

where $\bar{E}_0 = \bar{P}_1 - \bar{P}_0$ and $\bar{E}_1 = \bar{P}_2 - \bar{P}_0$, and $\bar{P}_0$, $\bar{P}_1$ and $\bar{P}_2$ are vectors that define the plane. s and r are parameters, defined in $\Re$.

A point inside a cone must fulfil the equation [88]

$$(\bar{A} \cdot (\bar{X}(s,r) - \bar{V}))^2 - \gamma^2 \|\bar{X}(s,r) - \bar{V}\|^2 = 0, \tag{A.2}$$

where $\bar{A}$ is the direction of the axis, $\bar{V}$ is the vertex of the cone, and $\gamma = cos(\alpha)$ where α is the opening angle of the cone. Equation (A.2) actually describes a double cone, but the point of intersection is on the side of the plane $\bar{A} \cdot (\bar{X} - \bar{V}) = 0$ to which $\bar{A}$ points. Furthermore, it is assumed that the cone is acute ($\alpha \in [0, \pi]$).

Inserting Equation (A.1) into Equation (A.2), we get

$$\begin{aligned}
&(\bar{A} \cdot (\bar{P}_0 + s\bar{E}_0 + r\bar{E}_1 - \bar{V}))^2 - \gamma^2 \|\bar{P}_0 + s\bar{E}_0 + r\bar{E}_1 - \bar{V}\|^2 = \\
&(\bar{A} \cdot (\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1))^2 - \gamma^2 \|\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1\|^2 = \\
&(\bar{A} \cdot (\bar{P}_0 + s\bar{E}_0 + r\bar{E}_1 - \bar{V}))^2 - \gamma^2 (\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1) \cdot (\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1) = 0,
\end{aligned} \tag{A.3}$$

which can be re-organised as

$$\begin{aligned}
&[(\bar{A}\bar{E}_0)^2 - \gamma^2 \bar{E}_0 \bar{E}_0] s^2 + [(\bar{A}\bar{E}_1)^2 - \gamma^2 \bar{E}_1 \bar{E}_1] r^2 + \\
&+ (2\bar{E}_0 \bar{A} \cdot \bar{A}\bar{\Delta}_0 - 2\gamma^2 \bar{E}_0 \bar{\Delta}_0) s + (2\bar{E}_1 \bar{A} \cdot \bar{A}\bar{\Delta}_0 - 2\gamma^2 \bar{E}_1 \bar{\Delta}_0) r + \\
&+ (2\bar{E}_0 \bar{A} \cdot \bar{E}_1 \bar{A} - 2\gamma^2 \bar{E}_0 \bar{E}_1) sr + \bar{A}\bar{\Delta}_0 - \gamma^2 \|\bar{\Delta}_0\|^2 = 0.
\end{aligned} \tag{A.4}$$

This is a quadratic equation

$$a_s s^2 + b_s(r) s + c_s(r) = 0, \tag{A.5}$$

where

$$a_s = (\bar{A}\bar{E}_0)^2 - \gamma^2 \bar{E}_0 \bar{E}_0, \tag{A.6}$$

$$b_s(r) = [2\bar{E}_0 \bar{A} \cdot \bar{E}_1 \bar{A} - 2\gamma^2 \bar{E}_0 \bar{E}_1] r + 2\bar{E}_0 \bar{A} \cdot \bar{A}\bar{\Delta}_0 - 2\gamma^2 \bar{E}_0 \bar{\Delta}_0, \tag{A.7}$$

$$c_s(r) = [(\bar{A}\bar{E}_1)^2 - \gamma^2 \bar{E}_1 \bar{E}_1] r^2 + (2\bar{E}_1 \bar{A} \cdot \bar{A}\bar{\Delta}_0 - 2\gamma^2 \bar{E}_1 \bar{\Delta}_0) r + \tag{A.8}$$

$$+ \bar{A}\bar{\Delta}_0 - \gamma^2 \|\bar{\Delta}_0\|^2, \tag{A.9}$$

Equation (A.9) can be solved using the quadratic formula, which gives for s

$$s(r) = \frac{-b_s(r) \pm \sqrt{b_s(r)^2 - 4a_s c_s(r)}}{2a_s}, \tag{A.10}$$

The function above is real when the discriminant is greater than or equal to zero. This means that the range of r can be solved

$$\begin{gathered} b_{\mathrm{c}}(r)^2 - 4a_{\mathrm{c}}c_{\mathrm{c}}(r) = \\ = (b_{1\mathrm{c}} + b_{2\mathrm{c}}r)^2 - 4a_{\mathrm{c}}(c_{1\mathrm{c}} + c_{2\mathrm{c}}r + c_{3\mathrm{c}}r^2) = \\ = (b_{2\mathrm{c}} - 4a_{\mathrm{c}}c_{3\mathrm{c}})r^2 + (2b_{1\mathrm{c}}b_{2\mathrm{c}} - 4c_{2\mathrm{c}}a_{\mathrm{c}})r + b_{1\mathrm{c}}^2 - 4a_{\mathrm{c}}c_{1\mathrm{c}} = \\ = a_{\mathrm{r}}r^2 + b_{\mathrm{r}}r + c_{\mathrm{r}} \geq 0, \end{gathered} \tag{A.11}$$

which also can be solved using the quadratic formula. To summarize we have

$$s(r) = \frac{-b_{\mathrm{c}}(r) \pm \sqrt{b_{\mathrm{c}}(r)^2 - 4a_{\mathrm{c}}c_{\mathrm{c}}(r)}}{2a_{\mathrm{c}}}, \text{ where} \tag{A.12}$$

$$r \leq \left| \frac{-b_{\mathrm{r}}(r) \pm \sqrt{b_{\mathrm{r}}^2 - 4a_{\mathrm{r}}c_{\mathrm{r}}}}{2a_{\mathrm{r}}} \right|. \tag{A.13}$$

A least square fitting method, presented by Fitzgibbon *et al.* [89], is used to find the parameters of the ellipse. The method takes as input a number of points, minimum five, to which an elliptic curve is fitted. The parameters of the fitted ellipse are given as output. The five points on the ellipse are created using the ranges of s and t defined above. As the ellipse fitting method requires the points to be in 2D, the created points are rotated about origin to the xy-plane prior to being passed to the fitting procedure. The fitting method gives the centre point $\bar{P}_{\mathrm{c}}$ of the ellipse, the length v and w of the major and minor axis respectively, and the angle θ between the major axis and the x-axis.

An ellipse can be expressed in parametric form as

$$\bar{P} = \bar{P}_c + v\cos\phi \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} - w\sin\phi \begin{pmatrix} \sin\theta \\ \cos\theta \end{pmatrix}, \tag{A.14}$$
$$\text{where } 0 \leq \phi \leq 2\pi.$$

By finding the line from the centre of ellipse i to the centre of ellipse j, $\bar{l}_{ij} = \bar{P}_{\mathrm{c},i} + \rho(\bar{P}_{\mathrm{c},i} - \bar{P}_{\mathrm{c},i})$, it is possible to solve the parameters ϕ and ρ for the intersection point between the line and the ellipse. This is done by determining the angle between the line $\bar{l}_{ij}$ and the major axis of one of the ellipses, using the definition of the vector dot product, which directly gives the ϕ parameter. Then the trivial matter of calculating the distance from the centre to the intersection point remains. Last the distance between the centres are reduced by the radii of the ellipses, R_{ij} and R_{ji}, in the direction of the other ellipses centre, by the equation

$$D_{ij} = d_{ij} - R_{ij} - R_{ji}, \tag{A.15}$$

and inserted into the distance matrix **D**.

The whole algorithm is presented below.

Algorithm 3 `PlaneConeIntersect`

Input: Plane parameters, Parameters for two of cones
Calculate parameters for the s and r equations for both cones
Find the rotation matrix that rotates the plane to the xy-plane
for all Cones **do**
Randomly create five point on the ellipse
Rotate the points to the xy-plane
Fit an ellipse to these points, using the least square fitting method
Save the ellipse parameters
end for
Calculate the distance between the centres of the ellipses
Calculate the direction of the line from the centre of one of the ellipses to the centre of the other
Solve ϕ by calculating the angle between the major axis and the line
Solve the intersection point between the line and one of the ellipses and calculate the distance from the intersection point to its centre
Solve the intersection point between the line and the other ellipse and calculate the distance from the intersection point to its centre
Calculate the $D_{ij} = d_{ij} - R_{ij} - R_{ji}$
return The distance D_{ij}

Appendix B Closest Distance from a Cone to a Triangle

This section presents how to find the closest point that is on a given triangle as well as inside a given cone to the vertex of the cone. It should be noted that the intersection of a triangle and a cone is not necessarily an ellipse, as only a part of the triangle may intersect with the cone. The procedure first determines if the closest point inside the cone lies on one of the edges of the triangle or along the curve of intersection. If not, the points not lying on any of the edges are evaluated.

In the following description it is assumed that the triangle and the cone intersect, so no intersection checks are necessary. In practice, this is evaluated using the algorithm presented by Eberly [88].

Triangle is defined by

$$\bar{X}(s,r) = \bar{P}_0 + s\bar{E}_0 + r\bar{E}_1, \tag{B.1}$$

where $\bar{X}$ is a point on the triangle, $\bar{E}_0 = \bar{P}_1 - \bar{P}_0$ and $\bar{E}_1 = \bar{P}_2 - \bar{P}_0$, so that

$$\begin{aligned} &s, r \geq 0, \\ &s + r \leq 1, \\ &s, r \in \Re. \end{aligned} \tag{B.2}$$

The distance from the vertex of the cone, $\bar{V}$, to a point on the triangle is given by

$$d = \|\bar{V} - \bar{X}(s,r)\|, \tag{B.3}$$

but without loss of generality, we can instead minimize

$$d^2 = \|\bar{V} - \bar{X}(s,r)\|^2, \tag{B.4}$$

to remove the need of square rooting and to simplify the calculation. This is the objective function of the optimization problem.

It is possible that the point on a triangle closest to the vertex is not inside the cone. For this reason we need to add another constraint to the model. A point inside the cone must fulfil [88]

$$(\bar{A} \cdot (\bar{X}(s,r) - \bar{V}))^2 - \gamma^2 \|\bar{X}(s,r) - \bar{V}\|^2 = 0, \tag{B.5}$$

where $\bar{A}$ is the axis of the cone, and $\gamma = \cos\alpha$ where α is the opening angle of the cone. Equation (B.5) actually describes a double cone, but it is assumed here that the triangle is on the side of the plane $\bar{A} \cdot (\bar{X} - \bar{V}) = 0$ to which $\bar{A}$ points. Furthermore, it is assumed that the cone is acute ($\alpha \in [0, \pi]$).

In Appendix A it was shown that the intersection of a plane and a cone is an elliptic curve defined by the parameters

$$s_{1,2} = \frac{-b_s(r) \pm \sqrt{b_s(r)^2 - 4a_s c_s(r)}}{2a_s}, \tag{B.6}$$

where

$$a_s = (\bar{A}\bar{E}_0)^2 - \gamma^2 \bar{E}_0 \bar{E}_0, \tag{B.7}$$
$$b_s(r) = [2\bar{E}_0\bar{A} \cdot \bar{E}_1\bar{A} - 2\gamma^2 \bar{E}_0\bar{E}_1]r + 2\bar{E}_0\bar{A} \cdot \bar{A}\bar{\Delta}_0 - 2\gamma^2 \bar{E}_0\bar{\Delta}_0, \tag{B.8}$$
$$c_s(r) = [(\bar{A}\bar{E}_1)^2 - \gamma^2 \bar{E}_1\bar{E}_1]r^2 + (2\bar{E}_1\bar{A} \cdot \bar{A}\bar{\Delta}_0 - 2\gamma^2 \bar{E}_1\bar{\Delta}_0)r + \tag{B.9}$$
$$+ \bar{A}\bar{\Delta}_0 - \gamma^2 \|\bar{\Delta}_0\|^2. \tag{B.10}$$

Furthermore, by solving r for when the discriminant of Equation (B.6) is non-negative, the range of allowed r-values is found, and the curve of intersection between the triangle and the cone can be determined.

Continuing from the objective function Equation (B.4), and inserting Equation (B.1) we get

$$\begin{aligned} d^2 =& \|\bar{X}(s,r) - \bar{V}\|^2 \\ =& \|\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1\|^2 \\ =& (\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1) \cdot (\bar{\Delta}_0 + s\bar{E}_0 + r\bar{E}_1) \\ =& \bar{E}_0\bar{E}_0 s^2 + 2\bar{\Delta}_0\bar{E}_0 s + 2\bar{E}_0\bar{E}_1 sr + 2\bar{\Delta}_0\bar{E}_1 r + \bar{E}_1\bar{E}_1 r^2 + \bar{\Delta}_0\bar{\Delta}_0, \end{aligned} \tag{B.11}$$

which can be minimized using Karush-Kuhn-Tucker (KTT) conditions,

$$\nabla f(x) + \sum_{i=1}^{m} \mu_i \nabla g_i(x) + \sum_{j=1}^{l} \lambda_j \nabla h_j(x) = 0, \tag{B.12}$$
$$g_i(x) \leq 0, \ i = 1, \ldots, m, \tag{B.13}$$
$$h_j(x) = 0, \ j = 1, \ldots, l, \tag{B.14}$$
$$\mu_i \geq 0, \ i = 1, \ldots, m, \tag{B.15}$$
$$\mu_i g_i(x) = 0, \ i = 1, \ldots, m, \tag{B.16}$$

where ∇ is vector differential operator. Applying this on Equation (B.11) and combining the KTT constraints with the constraints in Equation (B.2) and (B.6) we get

$$f(s,r) = \bar{E}_0\bar{E}_0 s^2 + 2\bar{\Delta}_0\bar{E}_0 s + 2\bar{E}_0\bar{E}_1 sr + 2\bar{\Delta}_0\bar{E}_1 r + \bar{E}_1\bar{E}_1 r^2, \tag{B.17}$$
$$g_1(x) = r, \tag{B.18}$$
$$g_2(x) = -s, \tag{B.19}$$
$$g_3(x) = s + r - 1, \tag{B.20}$$
$$g_4(x) = s - s_1, \tag{B.21}$$
$$g_5(x) = s_2 - s, \tag{B.22}$$

From this model, we get by separating into components,

$$2\bar{E}_0\bar{E}_0 s + 2\bar{\Delta}_0\bar{E}_0 + 2\bar{E}_0\bar{E}_1 r - \mu_2 + \mu_3 + \mu_4 - \mu_5 = 0, \tag{B.23a}$$
$$\mu_1 r = 0, \tag{B.23b}$$
$$\mu_2 s = 0, \tag{B.23c}$$
$$\mu_3(s + r - 1) = 0, \tag{B.23d}$$
$$\mu_4(s - s_1) = 0, \tag{B.23e}$$
$$\mu_5(s_2 - s) = 0. \tag{B.23f}$$

Based on these equations it is possible to formulate a number of cases.

Case 1 Both $s = 0$ and $r = 0$, meaning that the point closest to the vertex is $\bar{P}_0$. For this case to be true the following constraints needs to be fulfilled

$$\bar{\Delta}_0\bar{E}_1 \geq 0,$$
$$\bar{\Delta}_0\bar{E}_0 \geq 0,$$

Case 2 Only $s = 0$ while $r > 0$, which means that the closest point is on the edge starting from $\bar{P}_0$ and ending in $\bar{P}_2$. The constraint that needs to be fulfilled is

$$-\frac{2\bar{E}_0\bar{E}_1 \cdot \bar{\Delta}_0\bar{E}_1}{\bar{E}_1\bar{E}_1} + 2\bar{\Delta}_0\bar{E}_1 \geq 0,$$

which means that

$$r = -\frac{\bar{\Delta}_0\bar{E}_1}{\bar{E}_1\bar{E}_1},$$

is the value of parameter r at the point closest to the vertex.

Case 3 The point lies on the edge starting form P_0 and ending in P_1, so that $r = 0$ while $s > 0$, which requires that

$$-\frac{2\bar{E}_0\bar{E}_1 \cdot \bar{\Delta}_0\bar{E}_0}{\bar{E}_0\bar{E}_0} + 2\bar{\Delta}_0\bar{E}_0 \geq 0.$$

This means that the optimal value for s is

$$s = -\frac{\bar{\Delta}_0\bar{E}_0}{\bar{E}_0\bar{E}_0}.$$

Case 4 If both $s > 0$ and $r > 0$, the closest point lies on the edge of the cone. In this case the optima values for s and r are

$$s = \frac{\bar{E}_0\bar{E}_1 \cdot \bar{\Delta}_0\bar{E}_1 - \bar{E}_1\bar{E}_1 \cdot \bar{\Delta}_0\bar{E}_0}{\bar{E}_1\bar{E}_1 \cdot \bar{E}_0\bar{E}_0 - (\bar{E}_0\bar{E}_1)^2},$$
$$r = \frac{\bar{E}_0\bar{E}_1 \cdot \bar{\Delta}_0\bar{E}_0 - \bar{E}_0\bar{E}_0 \cdot \bar{\Delta}_0\bar{E}_1}{\bar{E}_1\bar{E}_1 \cdot \bar{E}_0\bar{E}_0 - (\bar{E}_0\bar{E}_1)^2}.$$

Case 5 If the closest point lies on the third edge of the triangle, between point P_1 and point P_2, then $s + r = 1$. This requires that

$$\bar{E}_0\bar{E}_0 s + \bar{\Delta}_0\bar{E}_1 + \bar{E}_0\bar{E}_1 r \leq 0.$$

In this case the optimal values for s and r are

$$s = \frac{\bar{E}_1\bar{E}_1 + \bar{\Delta}_0\bar{E}_1 - \bar{E}_0\bar{E}_1 - \bar{\Delta}_0\bar{E}_0}{\bar{E}_0\bar{E}_0 - 2\bar{E}_0\bar{E}_1 - \bar{E}_1\bar{E}_1},$$
$$r = \frac{\bar{E}_0\bar{E}_0 + \bar{\Delta}_0\bar{E}_0 - \bar{E}_0\bar{E}_1 - \bar{\Delta}_0\bar{E}_1}{\bar{E}_0\bar{E}_0 - 2\bar{E}_0\bar{E}_1 - \bar{E}_1\bar{E}_1}.$$

The values of the s and r parameters found in the cases above are also required to fulfil the constraints in Equations (B.2) and (B.6). Furthermore, the points are required to be on the correct side of the double cone by satisfying the equation

$$\bar{A} \cdot \left(\frac{(\bar{X}(s,r) - \bar{V})}{\|\bar{X}(s,r) - \bar{V}\|} \right) = \cos\theta. \tag{B.24}$$

If the closest point is not on the edges of the triangle or the ellipse, an inspection of the points within the edges is required. This is done by evaluating significant points in which $s_{1,2} \geq 0$ and $s_{1,2} + r \geq 0$ and $s_{1,2} + r \leq 1$. By assigning $s = s_1(r)$ and $s = s_2(r)$, and denoting $s_{1,2}(r)$ as $s_{1,2}$ for simplicity, and inserting into the objective function, Equation (B.11), the function becomes a single variable function

$$f_{1,2}(r) = \bar{E}_0\bar{E}_0 s_{1,2}^2 + 2\bar{\Delta}_0\bar{E}_0 s_{1,2} + 2\bar{E}_0\bar{E}_1 s_{1,2} r + 2\bar{\Delta}_0\bar{E}_1 r + \bar{E}_1\bar{E}_1 r^2, \tag{B.25}$$

which can be differentiated

$$\begin{aligned} \frac{df_{1,2}}{dr} &= 2\bar{E}_0\bar{E}_0 s'_{1,2} s_{1,2} + 2\bar{\Delta}_0\bar{E}_0 s'_{1,2} + 2\bar{E}_0\bar{E}_1 s_{1,2} + \\ &+ 2\bar{E}_0\bar{E}_1 s'_{1,2} r + 2\bar{\Delta}_0\bar{E}_1 + 2\bar{E}_1\bar{E}_1 r, \end{aligned} \tag{B.26}$$

where $s'_{1,2} = \frac{ds_{1,2}(r)}{dr}$. By finding the root(s) of the derivative, by using, for example, a bisection algorithm, the minimum point can be found, given that it is within the allowed range of r values. Remember that there exists two s-values and, therefore, two f functions to minimize. As the function only is real where the discriminant of Equation (B.6) is

non-negative, the range covered in the bisection algorithm only needs to span this region. The discriminant has, being a second order polynomial of r, two roots, which needs to be taken into account in selecting the range for the bisection algorithm. Furthermore, the constraints $s_{1,2} \geq 0$ and $r + s_{1,2} \leq 1$ can further limit the range.

To summarize, several significant points are evaluated against all constraints set on a feasible point: the roots of both $\frac{df_1}{dr}$ and $\frac{df_2}{dr}$ as well as the points where $s_{1,2} = 0$, and $s_{1,2} + r = 1$. Only after the feasibility of the points has been determined, their respective distances to the vertex of the cone can be calculated and compared. Last, the minimum distance of these points is returned.

Appendix C Coverage Problem Algorithms

This section presents, as pseudo code, the algorithms implemented in the population module to solve the coverage problems. A more general description of the algorithms can be found in Sections 6.1.1, 6.2.1 and 6.3.1, and a more detailed description of how they were implemented as a part of the automatic treatment planning algorithm in Section 9.2

Greedy Algorithm

The greedy algorithm creates iteratively a subset of cells, selecting the cells that, at each step, seems to maximize the objective value. The algorithm updates the set of feasible cells after each selection. The BUC algorithm runs two algorithms, one that selects cells with the highest weighted coverage (`GreedyBUC_maxCov`) and another that selects the cell with the highest weighted coverage-to-cost ratio (`GreedyBUC_maxRatio`) until the budget constraint or the overlap constraint does not allow more cells to be added. The algorithm then returns the better subset of the two. The MCUCT algorithm selects cells with the lowest cost-to-weighted coverage ratio until the threshold constraint is satisfied. If the algorithm encounters a situation where no more cells can be added without causing overlapping but the threshold criterion still is not satisfied, the algorithm fails.

The GRASP algorithm calls the greedy algorithm in its construction phase. The algorithm issues a selection parameter $\alpha \in [0, 1]$ that controls the size of the subset it selects cells from at each step. For the greedy algorithm $\beta = 1.0$.

Algorithm 4 `GreedyBUC`

Input: CoverMatrix, Weights, Costs, Budget, β
Output: Set of cells
  if No β defined **then**
    $\beta = 1.0$
  end if
  $optim1 =$ `GreedyBUC_maxCov`$(CoverMatrix, Weights, Costs, Budget, \beta)$
  $optim2 =$ `GreedyBUC_maxRatio`$(CoverMatrix, Weights, Costs, Budget, \beta)$
  Select the better of the two solutions
  return Set of cells

Algorithm 5 `GreedyBUC_maxCov`

Input: CoverMatrix, Weights, Costs, Budget
Output: Set of cells
Remove the cells that have a cost higher than the budget
while Budget not exceeded **do**
Get subset of cells that will not cause the budget to be exceeded or an overlap to occur
if List is empty **then** {No more cells can be added}
return Set of cells
else
Select one cell at random from the subset with $coverage \geq \beta \cdot coverage_{\text{best}}$
end if
end while
return Set of cells

Algorithm 6 `GreedyBUC_maxratio`

Input: CoverMatrix, Weights, Costs, Budget
Output: Set of cells
while Budget not exceeded **do**
Get subset cells that will not cause the budget to be exceeded or an overlap to occur
if List is empty **then** {No more cells can be added}
return Set of cells
else
Select one cell at random from the subset with $ratio \geq \beta \cdot ratio_{\text{best}}$
end if
end while
return Set of cells

Algorithm 7 `GreedyMCUCT`

Input: CoverMatrix, Weights, Costs, Threshold
Output: Set of cells
if No β defined **then**
$\beta = 1.0$
end if
while Threshold not exceeded **do**
Get subset of cells that will not cause an overlap to occur
if List is empty **then**
No more cells can be added and the algorithm fails
else
Select one cell at random from the subset with $ratio \leq (1 - \beta) \cdot ratio_{\text{best}}$
end if
end while
return Set of cells

GRASP Algorithm

The GRASP algorithms are related to the greedy algorithms, but do not select the cells with the best objective value, but selects randomly from all cells with an objective value $\beta * 100\%$ of the highest value. After a solution has been created, a local search is run, with the aim to improve the solution further. These two steps are repeated a number of times and the best solution produced is chosen as the final result.

Algorithm 8 `GRASP_BUC`

Input: CoverMatrix, Weights, Costs, Budget, Iterations, β
Output: Set of cells
  for all Iterations **do**
    $optim_i =$ `GreedyBUC`(Weights, Costs, Budget, β)
    $optim_i =$ `LocalSearch_BUC`($optim_i$)
    if Goodness of $optim_i$ is better than the currently best solution **then**
      Update the currently best solution to $optim_i$
    end if
  end for
  return Set of cells

Algorithm 9 `Localsearch_BUC`

Input: $optim$, CoverMatrix, Weights, Costs, Budget
Output: Set of cells
  for all Cells in optim **do**
    Remove cell j from $optim$
    for all Cells not in $optim$ **do**
      Put new cell in instead of removed cell
      Check feasibility and goodness
      if New set is feasible and has better goodness than currently best set **then**
        Update currently best set with the new set
      end if
    end for
  end for
  return Set of cells

Algorithm 10 `GRASP_MCUCT`

Input: CoverMatrix, Weights, Costs, Threshold, Iterations, β
Output: Set of cells
  for all Iterations **do**
    $optim_i =$ `Greedy_MCUCT`(Weights, Costs, Threshold, β)
    $optim_i =$ `LocalSearch_MCUCT`($optim_i$)
    if Goodness of $optim_i$ is better than the currently best solution **then**
      Update the currently best solution to $optim_i$
    end if
  end for
  return Set of cells

Algorithm 11 `Localsearch_MCUCT`

Input: $optim$, CoverMatrix, Weights, Costs, Threshold
Output: Set of cells
  for all cells in $optim$ **do**
    Remove cell j from $optim$
    for all Cells not in $optim$ **do**
      Put new cell in instead of removed cell
      Check feasibility and goodness
      if New set is feasible and has better goodness than currently best set **then**
        Update currently best set with the new set
      end if
    end for
  end for
  return Set of cells

Genetic Algorithm

The implemented genetic algorithm is based on the work of Mitchell [77]. To improve the algorithms ability to deal with the overlap and budget/threshold constraints, penalties were issued on chromosomes that did not fulfil all constraints. Moreover, a heuristics repair operator was implemented to fix newly constructed infeasible chromosomes into feasible ones. Elitism was used to assure that the best chromosomes are not destroyed by the evolution.

Algorithm 12 `GA_coverage`

Input: CoverMatrix, Weights, Costs, Threshold, Budget
Output: Set of cells

Create initial population of m chromosomes, either randomly or using a GRASP or greedy-like heuristic
for all generations **do**
{Evaluate the fitness of the population} {if BMC problem}
fit= `EvalFitness_BUC`$(Population, CoverMatrix, Weights, Costs, Budget)$
{if MCUCT problem}
fit= `EvalFitness_MCUCT`$(Population, CoverMatrix, Weights, Costs, Threshold)$

Scale the fitness values using sigma scaling
Select the n best chromosome for elitism
Select the $m-n$ chromosomes that will act as parents using roulette wheel selection

Use crossover to create $m-n$ offspring, either using one point or uniform crossover

Mutate the offspring at random
{Use Heuristic repair to produce a population consisting of feasible chromosomes}
{if BMC problem}
$pop =$ `HeurisitcRepir_BUC`$(Population, CoverMatrix, Weights, Costs, Budget)$
{if MCUCT problem}
$pop =$ `HeurisitcRepir_MCUCT`$(Population, CoverMatrix, Weights, Costs, Threshold)$

Add the n elite chromosomes
end for
return Set of cells

Algorithm 13 `FitnessEval_BUC`

Input: Population, CoverMatrix, Weights, Costs, Budget
Output: Fitness values

for all Chromosomes in population **do**
if Overlapping or budget exceeded **then**
$FitnessVal(i) = 0$
else
Evaluate $Coverage$ for chromosome i
$FitnessVal(i) = Coverage$
end if
end for
return $FitnessVal$

Algorithm 14 `FitnessEval_MCUCT`

Input: Population, CoverMatrix, Weights, Costs, Threshold
Output: Fitness values
for all Chromosomes in population **do**
if Overlapping or threshold not exceeded **then**
$FitnessVal(i) = 0$
else
Evaluate $Cost$ for chromosome i
$FitnessVal(i) = 1/Cost$
end if
end for
return Fitness values

Algorithm 15 `HeuristicRepair_BUC`

Input: Population, CoverMatrix, Weights, Costs, Budget
Output: Feasible population
for all Chromosomes in population **do**
while There is overlapping or budget is exceeded **do**
Remove the cell that, when removed, maximizes the weighted coverage-to-cost ratio of the remaining chromosome
end while
end for
return Feasible population

Algorithm 16 `HeuristicRepair_MCUCT`

Input: Population, CoverMatrix, Weights, Costs, Threshold
Output: Feasible population
for all Chromosomes in population **do**
while There is overlapping **do**
Remove the cell that, when removed, minimizes the cost-to-profit-ratio of the remaining chromosome
end while
while The threshold is not exceeded **do**
Add the cell that minimizes the cost-to-weighted coverage ratio of the chromosome
end while
end for
return Feasible population

Appendix D Routing Problem Algorithms

The algorithms used for solving the longest path problem (LPP) are presented below as pseudo code. A more general description of the algorithms can be found in Sections 6.1.2, 6.2.2, and 6.3.2, and a more detailed description of how they were implemented in the automatic treatment planning algorithm is given in Section 9.3

Farthest Neighbour Algorithm

The farthest neighbour algorithm is a greedy-like algorithm for solving the LPP. It is a heuristics that iteratively adds nodes to the path that is the furthest distance from the current node and has not yet been chosen.

Algorithm 17 `GreedyLPP`

Input: DistanceMatrix
Output: A path
    Select a starting node at random
    while Cell left to select **do**
        Select the node furthest away from the current node that has not already been chosen
    end while
5: **return** The constructed path

Genetic Algorithm

The genetic algorithm used for solving the LPP is based on the Enhanced Genetic Algorithm, presented by Yang and Stacey [90], improved with a judgement day operator, as proposed by Kureichick *et al.* [83], and 2-opt local search operator as presented by P. Larrañaga *et al.* [79]. The only parameters the algorithm takes as input are the population size as well as the parameters that control when the judgement day and local search operators are called. The algorithm utilizes a greedy crossover operator, and combines the crossover and mutation operators with inherent elitism.

Algorithm 18 `GA_LPP`

Input: DistanceMatrix
Output: A path
Set generation size and number of generation
Initialize a population randomly or by utilizing some heuristics
for all Generations **do**
while Not all parents have been selected once **do**
Randomly select two parents that have not been selected before
From two children by using greedy crossover
Evaluate the fitness of the children
Select the two fittest chromosomes of the two parents and two children
if The two fittest have the same fitness **then**
Keep one, and substitute the other for a randomly generated chromosome
end if
Pass the two new individuals forward to the next generation
end while
end for
return The best path from the last generation

Branch-and-Bound Algorithm

The BB algorithm for LPP does not require any subpath elimination constraints, which would be needed if the problem would be solved using regular IP solving methods. Instead, it relaxes the problem so that subpaths are not prohibited, and solves the corresponding *assignment problem*. If the found solution does not contain subpaths, the solution is the optimal and the algorithm exits. If, however, the path does contain subpaths, the algorithm commences the branching and prohibits, one after the other, the arcs in the subpaths, until a feasible solution is found. The algorithm is initiated by finding a good solution using the FN heuristics. This path sets the bound for the problem; solutions found by the BB algorithm with lengths shorter than the bound are not interesting and are not required to be investigated further. Using this method of initialisation can, in some cases, provide an increase in computational efficiency, as the number of branches that are to be investigated is better constrained. The BB algorithm, being deterministic, guarantees to find the optimal solution for the LPP problem.

Algorithm 19 `BB_LPP`

Input: DistanceMatrix
Output: Optimal path

```
    {STEP 1}
    Construct an initial feasible path by some heuristics, e.g. FN, to form an upper bound

    {STEP 2}
    Solve the assignment problem.
 5: if The solution has no subtours then {It is the optimal solution}
      return Optimal path
    else
      {STEP 3}
      Choose the subtour with least nodes
10:   {STEP 4}
      for all Arcs in the subtour do
        Add a constraint that prohibits the arc
        {STEP 5}
        Solve the Assignment problem with the added constraint
15:     Evaluate the objective value
        {STEP 6}
        if Objective value is greater than bound, but subtours exists then {Branch to the
        next level}
          Go to STEP 3
        else if Objective value is greater than bound, and no subtours then {Best feasible
        solution until now}
20:       Update the best solution and the bound.
          Continue for-loop
        else {Not good solution}
          Continue for-loop
        end if
25:   end for
    end if
    return Optimal path
```