Aalto University
School of Science

Master’s Programme in Mathematics and Operations Research

Optimization of Radar and Wind Farm Placement Using Computationally Intensive Simulations

Robin Hagnäs

Master’s Thesis
2025

Aalto University
School of Science

Author	Robin Hagnäs
Title	Optimization of Radar and Wind Farm Placement Using Computationally Intensive Simulations
Degree programme	Mathematics and Operations Research
Major	Systems and Operations Research
Supervisor	Prof. Kai Virtanen
Advisor	Prof. Kai Virtanen
Date 26 May 2025	**Number of pages** 60 **Language** English

Abstract

Wind farms have adverse effects on air surveillance radars. As no technical solution completely removes these adverse effects, this thesis studies finding the optimal combination of radar and wind farm placements to minimize the impact.

The performance of radars and the impact of wind farms on their performance can be analyzed with simulation models. However, these simulations are computationally costly. There are many alternative ways to place radars and wind farms, and simulating the quality of one alternative takes minutes. Thus, identifying the optimal alternative by simulating each one is time-consuming.

To identify the optimal alternative faster, this thesis presents two problem-specific algorithms. They return optimal solutions if simple assumptions about the underlying simulation models hold. These assumptions are based on the idea that radars improve air surveillance, wind farms degrade radar performance, and these impacts are limited to geographical regions. Numerical experiments show that the algorithms terminate in hours, when the simulation of every alternative would take years. Moreover, they returned near-optimal alternatives, despite slight violations of the assumptions.

The algorithms use the assumptions to compute alternative specific upper bounds for air surveillance quality. Air surveillance quality represents how well surveillance objectives are met. The algorithms work by iteratively identifying an alternative with a high upper bound and evaluating it using simulation. Once no upper bound exceeds an alternative's simulated air surveillance quality, the alternative is considered optimal and returned. The two algorithms differ in that the first introduces a basic approach, while the second builds on it by providing more efficient search strategies for finding alternatives with high upper bounds.

This thesis serves as a case study of developing problem-specific surrogate-assisted optimization algorithms. These algorithms facilitate the coexistence of air surveillance and wind farms, and maximize the utilization of air surveillance radars. Although this thesis focused on ground-based and monostatic radars, similar algorithms could be used with other radars, such as passive and airborne radars.

Keywords air surveillance, simulation, surrogate-assisted optimization, radar, wind power

Aalto-yliopisto
Perustieteiden
korkeakoulu

Tekijä Robin Hagnäs

Työn nimi Tutka- ja tuulipuistosijoittelun optimointi laskennallisesti raskailla simulaatioilla

Koulutusohjelma Mathematics and Operations Research

Pääaine Systems and Operations Research

Työn valvoja Prof. Kai Virtanen

Työn ohjaaja Prof. Kai Virtanen

Päivämäärä 26.5.2025 **Sivumäärä** 60 **Kieli** englanti

Tiivistelmä

Tuulipuistot aiheuttavat haittavaikutuksia ilmavalvontatutkille. Koska haittojen täydelliseen postamiseen ei ole teknistä keinoa, tässä diplomityössä tutkitaan tuulipuistojen ja tutkien sijoittelun optimointia haittavaikutusten minimoimiseksi.

Tutkien suorituskykyä ja siihen kohdistuvia tuulipuistojen haittavaikutuksia voidaan arvioida simulaatiomalleilla, mutta ne ovat laskennallisesti raskaita. Kaikkien mahdollisten sijoitteluvaihtoehtojen simulointi optimaalisen ratkaisun löytämiseksi on erittäin hidasta suuren vaihtoehtomäärän vuoksi.

Tässä diplomityössä esitetään kaksi optimointialgoritmia, jotka nopeuttavat optimaalisen vaihtoehdon tunnistamista. Algoritmit palauttavat optimaalisen vaihtoehdon, jos yksinkertaiset oletukset pitävät paikkansa. Nämä oletukset perustuvat siihen, että tutkat parantavat ilmavalvonnan laatua, tuulipuistot heikentävät sitä, ja nämä vaikutukset ovat maantieteellisesti rajoittuneita. Numeeriset esimerkit osoittavat, että algoritmit konvergoituvat tunneissa, kun kaikkien vaihtoehtojen simulointi veisi ajallisesti useita vuosia. Lisäksi ne tuottavat lähes optimaalisia ratkaisuja, vaikka oletuksia rikottaisiin hieman.

Algoritmit hyödyntävät oletuksia, kun lasketaan vaihtoehtojen ylärajoja ilmavalvonnan laadulle. Algoritmien toimintaperiaate on iteratiivisesti löytää vaihtoehto, jolla on korkea yläraja, ja arvioida simuloiden tämän mahdollistama ilmavalvonnan laatu. Vaihtoehto on optimaalinen, kun minkään muun vaihtoehdon yläraja ei ylitä sen simuloitua ilmavalvonnan laatua. Työssä esitetyt kaksi algoritmia eroavat siten, että ensimmäinen perustuu yksinkertaiseen toimintaperiaatteeseen, kun taas toinen käyttää tehokkaampia hakumenetelmiä korkean ylärajan vaihtoehtojen löytämiseksi.

Tämä diplomityö toimii esimerkkitapauksena simulointi-optimointiin tarkoitetun ongelmakohtaisen apumalliavusteisen algoritmin kehittämisestä. Kehitettyjen algoritmien avulla löydetään ratkaisuja ilmavalvonnan ja tuulivoiman yhteensovittamiseksi ja maksimoidaan tutkien suorituskyky. Vaikka tässä työssä keskityttiin maasijoitteisiin monostaattisiin tutkiin, vastaavia algoritmeja voitaisiin hyödyntää myös muiden tutkien, kuten passiivisten ja ilmasijoitteisten tuktien, sijoittelun optimointiin.

Avainsanat ilmavalvonta, simulointi, apumalliavusteinen optimointi, tutka, tuulivoima

Aalto-universitetet
Högskolan för
teknikvetenskaper

Författare Robin Hagnäs

Titel Optimering av radar- och vindkraftsparksplacering med beräkningsintensiva simuleringar

Utbildningsprogram Mathematics and Operations Research

Huvudämne Systems and Operations Research

Övervakare Prof. Kai Virtanen

Handledare Prof. Kai Virtanen

Datum 26.5.2025 **Sidantal** 60 **Språk** engelska

Sammandrag

Vindkraftsparker orsakar störningar för luftbevakningsradarer. Eftersom det inte finns någon teknisk lösning som helt eliminerar störningarna, undersöker detta examensarbete optimering av radar- och vindkraftsparksplacering för att minimera dem.

Radarprestandan och vindkraftverkens påverkan på denna kan analyseras med simuleringsmodeller. Dessa simuleringar är dock beräkningsmässigt kostsamma. Det finns många möjliga kombinationer av radar- och vindkraftverksplaceringar, och att simulera luftbevakningsförmågan för ett enskilt alternativ tar flera minuter. Därför är det mycket tidskrävande att identifiera det optimala alternativet genom att simulera varje alternativ.

Detta arbete presenterar två algoritmer för att effektivera sökandet efter ett optimalt alternativ. Algoritmerna returnerar optimala lösningar om antaganden om simuleringsmodellerna uppfylls. Dessa antaganden baserar sig på att radarer förbättrar luftövervakningen, att vindkraftverk försämrar radarprestandan, och att dessa effekter är geografiskt begränsade. Numeriska experiment visar att algoritmerna konvergerar inom några timmar, medan simulering av alla alternativ skulle ta flera år, samt att algoritmerna ger nästintill optimala lösningar även när antagandena bryts.

Algoritmerna använder antagandena för att beräkna övre gränser för alternativens luftbevakningsförmågan. Algoritmerna bygger på idén att upprepat identifiera ett alternativ med en hög övre gräns och simulera dess luftbevakningsförmåga. När ingen övre gräns längre överskrider luftbevakningsförmågan hos ett simulerat alternativ, är detta alternativ optimalt. De två algoritmerna skiljer sig åt genom att den första presenterar deras idé i en enkel form, medan den andra vidareutvecklar den med effektivare sökstrategier för att hitta alternativ med höga övre gränser.

Detta examensarbete fungerar som en fallstudie i utvecklingen av problemspecifika optimeringsalgoritmer som utnyttjar surrogatmodeller. Med hjälp av dessa algoritmer är det möjligt att underlätta samexistensen mellan luftövervakning och vindkraft samt maximera radarprestandan. Även om arbetet fokuserar på markbaserade monostatiska radarer, kan liknande algoritmer tillämpas även på andra typer av radarer, såsom passiva och luftburna radarer.

Nyckelord luftbevakning, simulering, surrogatmodellbaserad optimering, radar, vindkraft

Preface

I would like to thank my supervisor, Professor Kai Virtanen, for his guidance, feedback, and support throughout this thesis project. I would also like to thank the individuals at Patria who were involved in this thesis project for their input and collaboration. Finally, I want to thank my colleagues, friends, and family for their help and encouragement.

Otaniemi, 26 May 2025

Robin Hagnäs

Contents

Abbreviations

ASR	Air surveillance requirement
ASQV	Air surveillance quality value
QUB	Quality upper bound
RRHC	Random restart hill climbing

1 Introduction

Wind farms are increasingly being constructed due to the growing demand for renewable energy. However, wind farms adversely affect air surveillance radars, reducing their capability of detecting and tracking targets (see, e.g., De la Vega et al., 2013). Surveilling targets such as hostile aircraft and missiles is critical for national security. Therefore, wind farms cannot be constructed if their adverse effects on air surveillance are severe (Huttunen et al., 2024; Joensuu et al., 2021). The adverse effects can be mitigated by various modifications to radars and wind farms, acquiring more radars, and optimizing radar and wind farm placement (De la Vega et al., 2013). However, no technical solution completely removes the effects, and acquiring more radars is expensive. Therefore, this thesis studies the optimization of radar and wind farm placement. The thesis considers ground-based and monostatic radars.

Optimization of radar and wind farm placement involves finding the combination of radar and wind farm placements that maximizes air surveillance quality. In this thesis, air surveillance quality (Virtanen, 2024) is measured by the fulfillment of air surveillance objectives. For instance, one objective could be to provide general situational awareness, and another to provide sufficient air surveillance to intercept hostile missiles. The computation of air surveillance quality requires evaluating the performance of radars with performance metrics in various geographical locations of 3D airspace. The performance metrics are obtained using an existing computational tool that utilizes accurate yet computationally intensive simulation models (see, Lahti, 2022; Virtanen, 2024). The metrics are used to evaluate air surveillance quality using a spatial multi-criteria value function (see, e.g., Harju et al., 2019).

Optimizing radar and wind farm placements with computationally intensive simulations is challenging, since simulating the performance of radar and wind farm placements is time-consuming. For instance, there are approximately 10^5 combinations of placing five radars and five wind farms into the feasible sites presented in Figure 1. Simulating the radars' performance in a single combination takes minutes, which means evaluating all 10^5 alternatives would require roughly a year of computation. Similarly, using evolutionary algorithms (see, e.g., Simon, 2013) would require the simulation of many alternatives, making them slow and unsuitable.

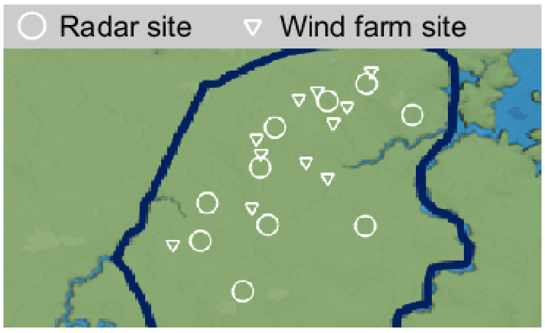


Figure 1: A fictional map with 10 radar sites and 11 wind farm sites. The background map is generated with Mapgen4 (Red Blob Games, 2018).

Solving optimization problems involving computationally expensive simulations has been studied in the disciplines of simulation optimization, derivative-free optimization, and blackbox optimization (see, Audet and Hare, 2017; Fu, 2015; Larson et al., 2019). These problems are often approached with surrogate models, also known as meta-models (see, e.g., Bartz-Beielstein and Zaefferer, 2017; Chugh et al., 2019). A surrogate model is a simpler and computationally cheaper approximation of the original expensive simulation. It is based on learning patterns from evaluated inputs and corresponding outputs of the original model. As computations with the surrogate model are faster than the expensive simulations, the surrogate model can be used to evaluate the quality of many alternatives quickly. This enables optimization algorithms to allocate computationally expensive evaluations to the most promising candidate solutions.

Many surrogate-assisted algorithms have been proposed for efficient optimization using expensive simulations in various domains (see, e.g., Jin, 2011; Shahriari et al., 2015). However, most are designed for continuous problems, where decision variables can take any value within a range. In the context of radar and wind farm placement, this would correspond to optimizing over continuous coordinates, meaning that radars or wind farms could be placed anywhere within a region, rather than selecting from a set of feasible locations. This thesis focuses on combinatorial problems where the goal is to choose the best combination of sites from a finite set of options. While some surrogate-assisted algorithm ideas for combinatorial problems exist (see, e.g., Baptista and Poloczek, 2018; Oh et al., 2019; Papalexopoulos et al., 2022), they are not designed to leverage domain-specific information of how radars and wind farms impact air surveillance.

As no tailored algorithm exists for the optimization of radar and wind farm placement, this thesis introduces the quality upper bound (QUB) algorithm and its extended variant. These algorithms leverage the insight that radars improve air surveillance, wind farms degrade radar performance, and these effects are confined to specific geographical regions. They combine the insight with previous simulation results to compute upper bounds for air surveillance quality. These upper bounds are referred to as QUBs. The proposed algorithms utilize the QUBs as their surrogate model. They find the optimal alternative by iteratively searching for an alternative with a high QUB and evaluating it using the expensive simulations. The purpose of the QUB algorithm is to present the idea of using QUBs to find the optimal alternative. The extended QUB algorithm improves this idea by utilizing more efficient methods for finding alternatives with a high QUB.

In this thesis, the algorithms are evaluated through theoretical analysis and numerical experiments. The theoretical analysis shows that the algorithms find optimal alternatives if the simulations adhere to assumptions based on the insight that radars improve air surveillance and wind farms adversely affect the radars. Numerical experiments demonstrate that the algorithms return near-optimal alternatives within hours, when the simulation of every alternative would take years, despite violations of the algorithms' assumptions.

Previous research has not addressed radar and wind farm placement optimization when the number of alternatives is too large to evaluate each one individually. Lahti

(2022) proposes a method for assessing air surveillance quality and wind farm impact, but it requires simulating all alternatives. This thesis builds on that approach and introduces algorithms to identify the optimal solution without simulating every alternative.

Radar and wind farm placement optimization with many alternatives have been addressed separately. Radar placement optimization has been studied with computationally cheap models without considering wind farms (see, e.g., Dhillon and Chakrabarty, 2003; Tema et al., 2024; Yang et al., 2015). In contrast, this thesis accounts for wind farms and uses computationally intensive models. Using computationally intensive models allows for the modeling of radar performance more accurately and leads to better optimization results. Wind farm placement has been optimized without considering its impact on air surveillance radars (see, e.g., Cranmer et al., 2018; Sunak et al., 2015). This thesis optimizes wind farm placement considering its impact on air surveillance.

This thesis is outlined as follows. First, the radar and wind farm placement problem is presented, including descriptions of decision alternatives and the computation of air surveillance quality. Next, the QUB algorithm and the computation of QUBs are introduced, followed by the analysis of the algorithm's performance. This is followed by the presentation and analysis of the extended QUB algorithm. The algorithms are then applied to example problems, demonstrating their efficiency. The discussions summarize the numerical experiments, describes the use of the algorithms, and offer ideas for future research. The thesis concludes by summarizing the findings.

2 The radar and wind farm placement problem

The radar and wind farm placement problem involves placing a constrained number of radars and wind farms to feasible sites, maximizing air surveillance quality. This quality measures the capability of the radars to fulfill multiple air surveillance objectives, considering the impact of the wind farms on the radars' performance. It is quantified by the air surveillance quality value (ASQV) presented by Virtanen (2024). This section introduces the problem by first outlining decision alternatives, then by describing the computation of the ASQV, and finally discussing the challenges of solving the problem.

2.1 Decision alternatives

The decision alternatives consist of combinations of positioning radars and wind farms, constrained by the number of feasible sites and available units. The decision alternatives are denoted by z_i, where the index $i = 1, ..., I$ represents a radar and wind farm placement. The total number of alternatives is denoted by I. To illustrate the decision alternatives, consider a scenario with 10 feasible radar sites and 11 feasible wind farm sites, where five radars and five wind farms are to be placed. Figure 2 shows the feasible sites, while Figure 3 provides an example of a decision alternative.

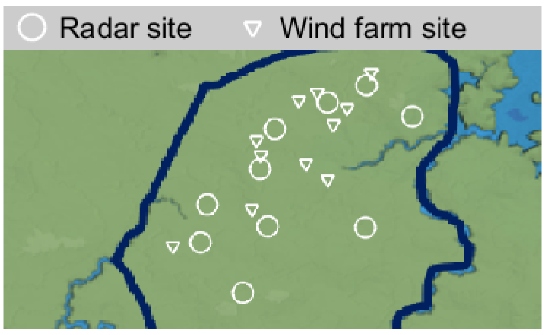


Figure 2: A scenario with 10 radar sites and 11 wind farm sites.

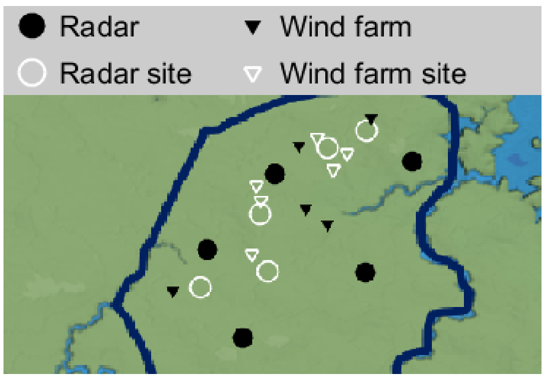


Figure 3: An example of a decision alternative.

The wind farm sites are assumed to be such that placing a wind farm there would be feasible if its impact does not hinder radars' capability to fulfill air surveillance objectives. Thus, if a wind farm is to be built at a site, a plan exists specifying how many wind turbines it will contain and where they will be located within the area. The factors impacting wind farm placement, such as economic viability, environmental concerns (see, e.g., Saidur et al., 2011), and public opinion (see, e.g., Ek and Persson, 2014) have been considered when selecting these sites and planning the individual wind turbine placements within the wind farm.

2.2 Computation of the air surveillance quality value

The computation of the ASQV requires two components: location-specific values of performance metrics, and air surveillance requirements (ASRs) (Virtanen, 2024). The performance metric values represent the radars' capability to detect and track targets, such as cruise missiles and fighter jets. They are computed with a computational tool (see, Lahti, 2022; Virtanen, 2024) considering factors such as terrain topography and the adverse effects of wind farms on the radars' performance. Thus, the tool provides realistic performance metric values, but their computation is time-consuming. For a more detailed description of the computational tool, see Virtanen (2024) and Lahti (2022). The performance metrics considered in this thesis are probability of detection, time between observations, and track accuracy. Table 1 shows descriptions of these metrics.

Table 1: The descriptions of the performance metrics.

Name	Description
Probability of detection	The probability that a target is detected in a location in the 3D-airspace
Time between observations	The average time passed between observations of the target
Track accuracy	The distance between the observed and actual position of the target

Each ASR represents an air surveillance objective. For instance, one objective could be to provide general situational awareness, and another to provide sufficient air surveillance to intercept hostile missiles. An ASR consists of a 3D-surveillance zone, target type, quality statement, and priority. The surveillance zone and target type define where the object of interest is located and what kind of object it is. The quality statement defines two thresholds for each performance metric: the worst and best quality levels. Values inferior to the worst quality level are considered insufficient, while those surpassing the best quality level are deemed sufficient. The priority specifies the importance of the ASR compared to other ASRs on an ordinal scale. For more details about ASRs, see Virtanen (2024).

The ASQV is calculated by first computing the performance metric values and mapping them to the $0-1$ interval with consequence value functions (for more details about value functions, see e.g., Malczewski and Rinner (2015)). The consequence value functions are ASR and performance metric-specific. They take into account the best and worst levels of the performance metrics given in the quality statement. In this thesis, the consequence value function for probability of detection is the increasing piecewise linear function

$$u_{j1}(p) = \begin{cases} 0\,, & p \leq \alpha_j \\ \frac{p-\alpha_j}{\phi_j-\alpha_j}, & \alpha_j < p < \phi_j \\ 1, & p \geq \phi_j \end{cases} \tag{1}$$

where j is the index of the ASR, the subscript 1 denotes that the consequence value function corresponds to probability of detection, p is the value of the performance metric, α_j is the worst quality level, and ϕ_j is the best quality level.

The performance metrics track accuracy and time between observations have a decreasing consequence value function, as a smaller value in these metrics is preferred over a higher value. The decreasing linear consequence value function is given by

$$u_{jk}(p) = \begin{cases} 1\,, & x \leq \phi_j \\ \frac{p-\alpha_j}{\phi_j-\alpha_j}, & \alpha_j < p < \phi_j \\ 0, & p \geq \alpha_j \end{cases} \tag{2}$$

where the subscript k denotes the performance metric that the function corresponds to. The subscript value 2 corresponds to time between observations and 3 to track accuracy.

The consequence value functions are utilized for computing the 2D fulfillment value of an ASR (Virtanen, 2024). In the framework of Virtanen (2024), the value is used to visualize where the ASRs are fulfilled. This value is not used for visualization in this thesis, but as an intermediate result in computing the ASQV and the QUBs in Section 3. Therefore, unlike in Virtanen (2024), the 2D fulfillment value of an ASR is not normalized to the $0 - 1$ interval in this thesis. The 2D fulfillment value of an ASR is an aggregation of the consequence values across the altitudes on the same 2D location. It is given by

$$V_j^{2D}(x, y; z^i) = \sum_{s \in S_j(x,y)} w_{js} \sum_{k=1}^{K} h_{jk} u_{jk}(z^i_{jk}(s)), \tag{3}$$

where $S_j(x, y)$ represents all the locations within the altitude interval of the 3D-surveillance zone at the coordinates x, y. s is a location in the 3D-surveillance zone, w_{js} are location specific weights, h_{jk} are performance metric specific weights. The weights w_{js} represent the importance of 3D locations, and the weights h_{jk} represent the importance of the performance metrics. The weights w_{js} sum to 1 when summed over all 3D locations, while the weights h_{jk} sum to 1 over all performance metrics. u_{jk} is the performance metric specific consequence value function, and $z^i_{jk}(s)$ is the value of performance metric k at the location s for radar and wind farm placement i.

The values of $z^i_{jk}(s)$ are computed with the computational tool. The maximum value of the 2D fulfillment value is $\sum_{s \in S_j(x,y)} w_{js}$, since the maximum value of $u_{jk}(z^i_{jk}(s))$ is 1.

The 2D fulfillment value of an ASR is used to compute the fulfillment of an ASR. The fulfillment of an ASR reflects how well the air surveillance objective that the ASR represents is fulfilled. It is given by

$$V_j(z^i) = \sum_{(x,y) \in S_j^{2D}} V_j^{2D}(x, y; z^i), \tag{4}$$

where $(x, y) \in S_j^{2D}$ means all unique (x, y)-coordinate pairs in the 3D-surveillance zone. Equation 4 appears to differ from the formulation of the fulfillment value of an ASR in Virtanen (2024). However, they are equivalent since

$$\begin{aligned} V_j(z^i) &= \sum_{(x,y) \in S_j} V_j^{2D}(x, y; z^i) && (5) \\ &= \sum_{(x,y) \in S_j^{2D}} \sum_{s \in S_j(x,y)} w_{js} \sum_{k=1}^{K} h_{jk} u_{jk}(z^i_{jk}(s)) && (6) \\ &= \sum_{s \in S_j} w_{js} \sum_{k=1}^{K} h_{jk} u_{jk}(z^i_{jk}(s)) && (7) \end{aligned}$$

and Equation 7 is the formulation for the fulfillment of an ASR presented in Virtanen (2024). This thesis uses the formulation of Equation 4 in the computation of QUBs in Section 3. This formulation is chosen because it is notationally more convenient in the context of the QUB computations

The fulfillment of the ASRs are used for computing the ASQV. The ASQV represents the overall fulfillment of the air surveillance objectives. It is a weighted sum of the fulfillments of all ASRs under consideration. These weights are calculated using the ordinal priorities of the ASRs with the centroid weights method (see, e.g., Ahn, 2011). The ASQV is given by

$$V(z^i) = \sum_{j=1}^{J} m_j V_j(z^i), \tag{8}$$

where m_j are ASR specific weights and J is the number of ASRs. The weights m_j quantify the relative importance of the ASRs, and they sum to 1.

2.3 Summary and challenges

To summarize, the goal of the radar and wind farm placement problem is to find the combination z^i of radar and wind farm placements that maximizes the ASQV given in Equation 8. An optimal alternative is one that has the maximal ASQV. The radar and wind farm placement problem is challenging since computing the ASQV for every alternative is time-consuming. The reasons for this are twofold. First, the number of decision alternatives increases rapidly with the number of feasible sites. For instance, there are approximately 10^5 combinations of placing five radars and five wind farms into the sites presented in Figure 2. In contrast, there are approximately 10^{15} ways of placing 10 radars and 20 wind farms into the sites of Figure 4.

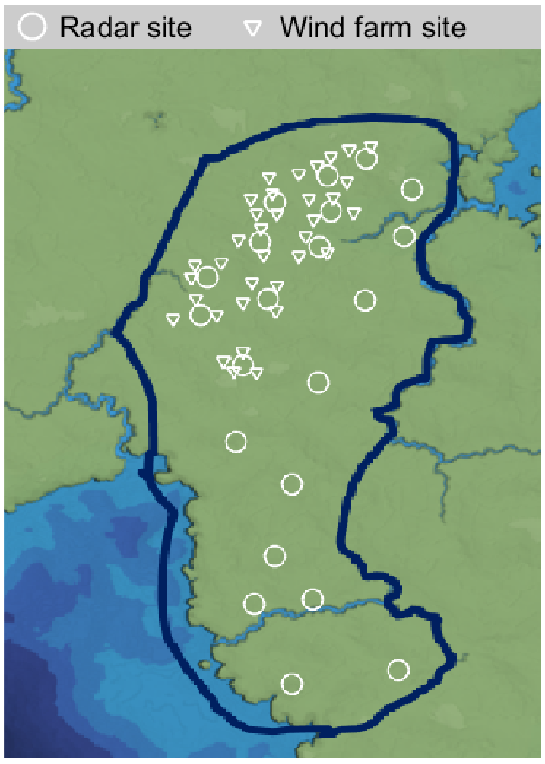


Figure 4: A scenario with 21 radar sites and 35 wind farm sites

Second, computing all performance metrics for a single combination of radars and wind farms takes minutes with the computational tool. Hence, computing them for 10^5 alternatives would take approximately a year, and for 10^{15} alternatives approximately 10^{10} years. The next two sections present the QUB algorithm and its extended variant to address the challenge of finding the optimal radar and wind farm placement combination without computing the ASQV for every alternative.

3 The quality upper bound algorithm

This section presents the QUB algorithm developed in this thesis to address the radar and wind farm placement problem presented in Section 2. As outlined in Section 2.3, the challenge of finding the radar and wind farm placement with the maximal ASQV using computationally intensive simulations is that computing the ASQV for every alternative is time-consuming. The reason for large computational requirements is that the evaluation of the ASQV with an existing computational tool (Lahti, 2022; Virtanen, 2024) and Equations 3, 4, and 8 takes minutes for a single alternative, and the total number of alternatives is large. The QUB algorithm solves the problem by finding the optimal alternative without determining the ASQV of each one.

The QUB algorithm finds the optimal alternative by utilizing QUBs. The QUB is an alternative-specific value greater than or equal to the ASQV that is faster to compute than the ASQV. An alternative is optimal if its ASQV is larger than the QUB of every other alternative. Figure 5 illustrates how the QUBs are utilized to verify that an alternative is optimal. In the figure, alternative 4 is optimal since if the ASQVs of alternatives 1, 2, and 3 were to be computed, they would be smaller than or equal to their QUBs.

The property of the QUB being greater than or equal to the ASQV relies on assumptions. These assumptions are based on the idea that placing a radar at a site generally increases air surveillance quality, that placing a wind farm near a radar degrades its performance, and that both effects are geographically constrained. In addition to the assumptions, the computation of a QUB for an alternative uses intermediate results obtained in computing the ASQVs of other alternatives. These intermediate results consist of 2D fulfillment values of ASRs.

The QUB algorithm finds the optimal alternative by iteratively computing the QUB of every alternative, and the ASQV of the alternative with the maximal QUB. As the ASQV of more alternatives is determined, the QUBs of the alternatives decrease, since there is more data for computing QUBs. Eventually, the algorithm stops when the ASQV of an alternative is greater than or equal to the QUB of every other alternative, as alternative 4 in Figure 5. Figure 6 demonstrates how the QUB algorithm finds the optimal alternative through an example, where the best alternative of five is found by computing the ASQV for three alternatives.

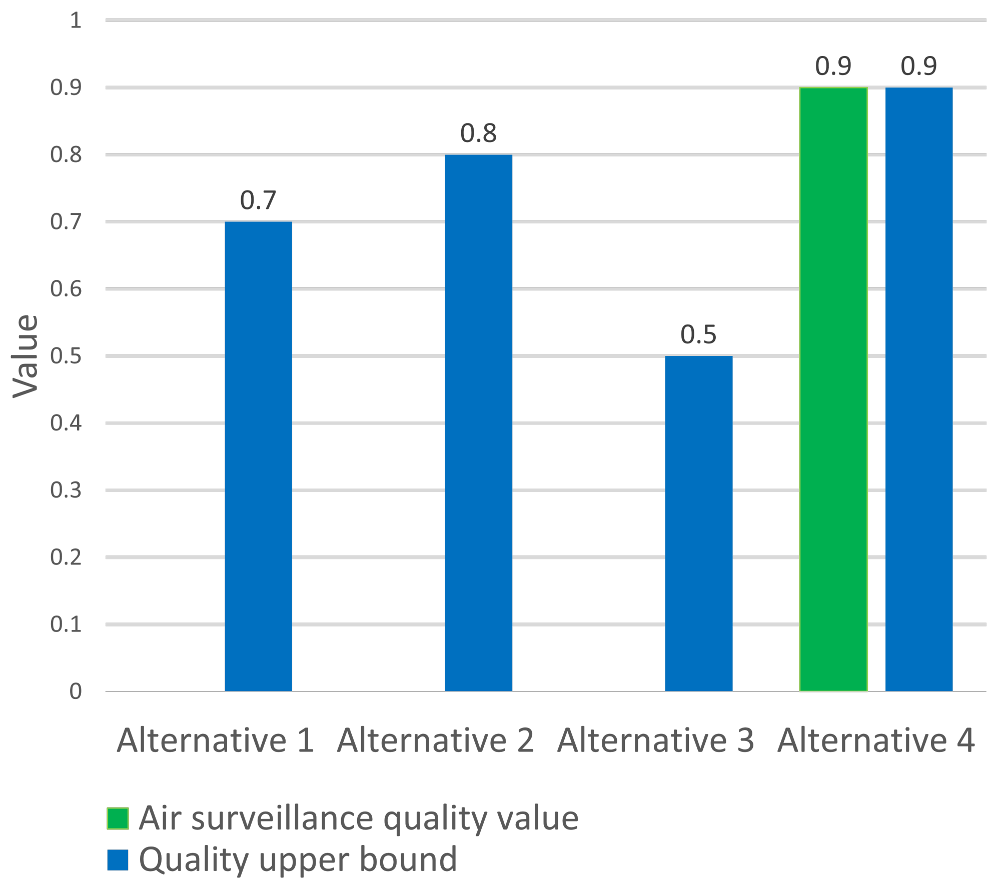
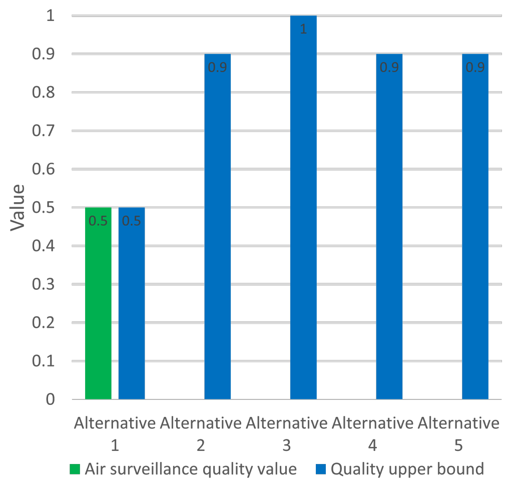


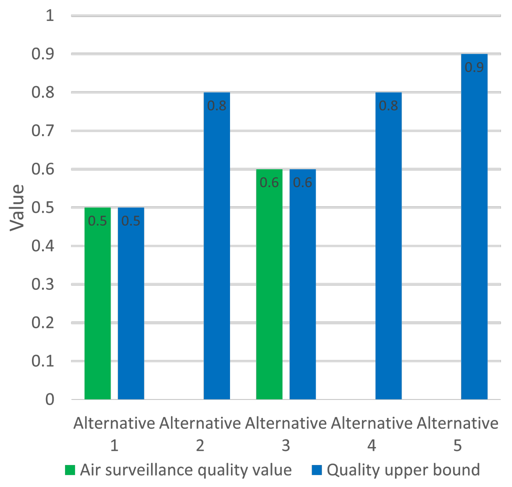
Figure 5: Alternative 4 is optimal since no QUB exceeds its ASQV.

The QUB algorithm finds the optimal alternative if the assumptions on which its computation is based hold. However, the assumptions can be violated in practice. Violations occur, for instance, if adding radars or removing wind farms decreases performance metric values, and their computation is stochastic. Violating the assumptions may cause QUBs of alternatives to be smaller than their computed ASQVs, which can lead to the algorithm stopping before the optimal alternative is found.

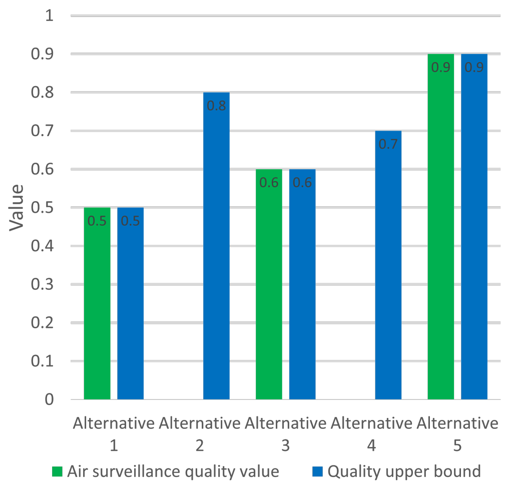
This section is structured as follows. First, the assumptions underlying the computation of QUBs are presented. Next, the computation of a QUB for an alternative is described. Then, the inputs, initialization, and iterative phase of the QUB algorithm are detailed. Following this, it is shown that the algorithm finds the optimal alternative when the assumptions, on which the computation of the QUBs is based, hold. Finally, the consequences of violating the assumptions and the limitations of the algorithm are discussed.



(a) The QUBs of all five alternatives after the ASQV of alternative 1 has been computed. Alternative 3 has the maximum QUB. Therefore, the ASQV of alternative 3 is computed in the next iteration.



(b) The QUBs after the ASQV of alternative 3 is computed. Alternative 5 has the maximum QUB. Therefore, the ASQV of alternative 5 is computed in the next iteration.



(c) The QUBs after computing the ASQV of alternative 5. No QUB exceeds the ASQV of alternative 5. Thus, alternative 5 is optimal.

Figure 6: An illustration of how the algorithm finds the best alternative from five in three iterations. Figure 6a shows the QUBs and computed ASQV at iteration 1. Figure 6b shows the QUBs and computed ASQVs at iteration 2. Figure 6c shows the QUBs and computed ASQVs at iteration 3.

3.1 Assumptions

The computation of QUBs assumes that the calculation of the performance metric values $z^i_{jk}(s)$ in Equation 3 is deterministic. That is, the computational tool produces identical output for identical input. Additionally, the computation of QUBs relies on four assumptions regarding adding and removing radars and wind farms. Next, these four assumptions are introduced.

The first two assumptions relate to the coverage of a radar. The coverage of a radar is defined as a radar and site-specific 2D region. For the assumptions in this section to hold, the coverage should include all 2D locations above which the radar affects performance metrics at some altitudes within the ASRs' surveillance zones. The QUB algorithm determines a coverage for each feasible radar and radar-site pair with a method presented in Section 3.3.

The first assumption considers the action of adding a radar, the 2D fulfillment value of an ASR (Equation 3), and the coverage of a radar.

Assumption 1. *Adding a radar increases or maintains the 2D fulfillment values of ASRs within the radar's coverage.*

The rationale behind Assumption 1 is that adding a radar does not diminish the capabilities of existing radars. The added radar can improve the 2D fulfillment values of ASRs by providing additional surveillance data. Consequently, the 2D fulfillment values of ASRs either remain unchanged or improve with the addition of a radar. Figure 7 illustrates Assumption 1. The 2D fulfillment value of an ASR at the 2D location depicted by the red square increases or remains unchanged due to the addition of radar 2.

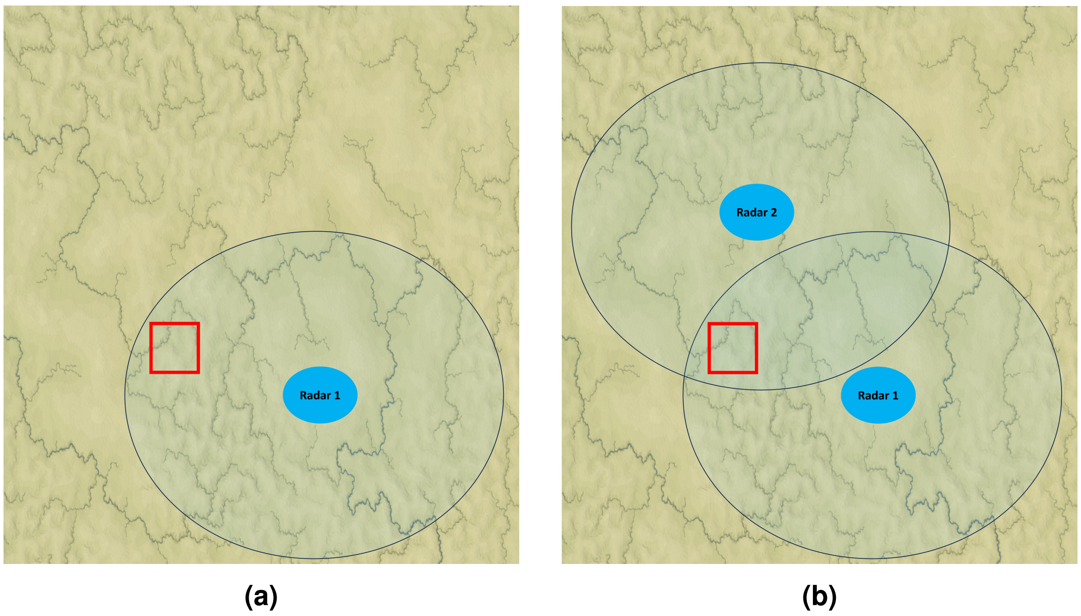


Figure 7: Figure 7a shows radar 1 and Figure 7b shows radar 1 at the same location along with radar 2. The 2D fulfillment value of an ASR at the location depicted by a red square in Figure 7b is greater than or equal to that in Figure 7a. Blue ellipses depict radars, and the shaded regions around them depict their coverages.

The second assumption is related to adding and removing radars that do not impact the 2D fulfillment values of ASRs in a 2D location.

Assumption 2. *Adding or removing a radar maintains 2D fulfillment values of ASRs outside the coverage of the radar.*

The rationale behind Assumption 2 is that if an added radar does not provide any surveillance data, it does not improve 2D fulfillment values of ASRs. Similarly, removing such a radar does not reduce the 2D fulfillment values of ASRs. As a result, the 2D fulfillment values of ASRs remain unchanged. Figure 8 illustrates Assumption 2 by showing a 2D location depicted by a red square, where the 2D-fulfillment value of an ASR remains unchanged with the addition and removal of a radar.

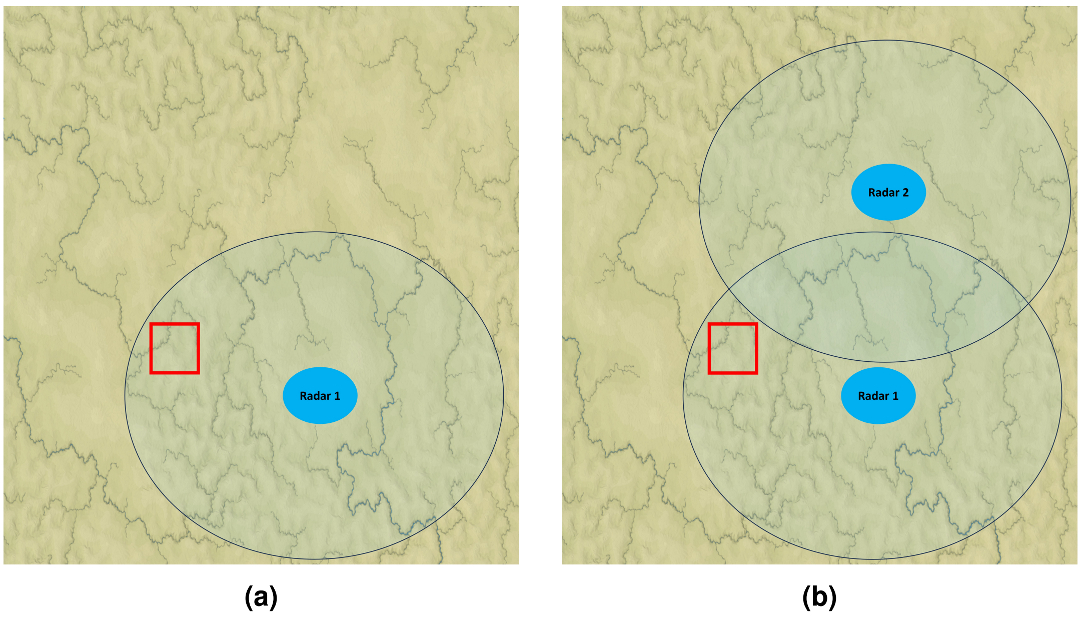


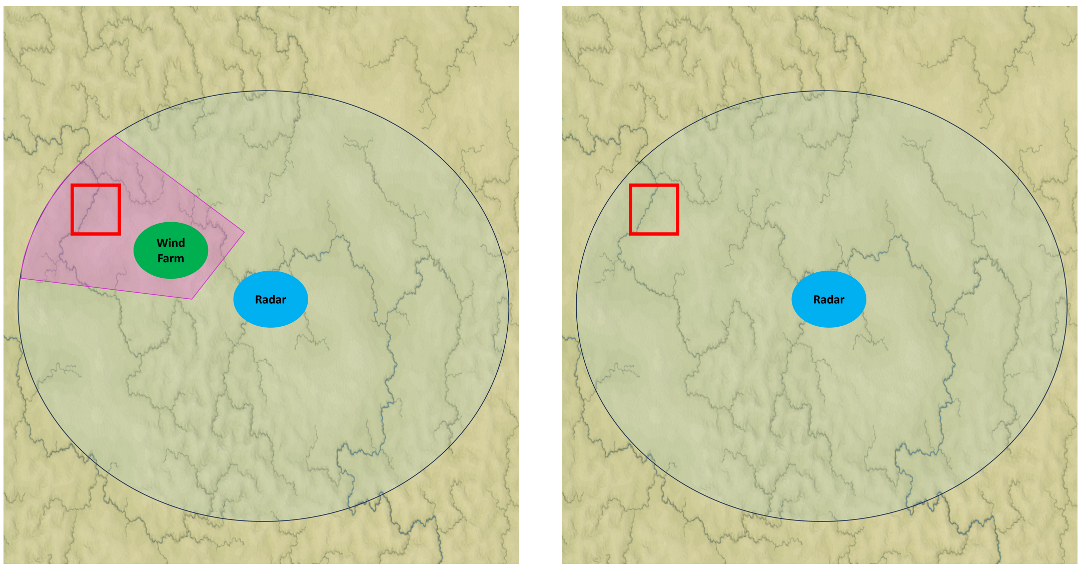
Figure 8: Figure 8a shows radar 1 and Figure 8b shows radar 1 at the same location along with radar 2. Radar 2 cannot detect targets in the 2D location marked by a red square. Thus, the 2D fulfillment value of an ASR at the red square is equal in the figures. Blue ellipses depict radars, and the shaded regions around them are their coverages.

The last two assumptions relate to adverse effect regions. An adverse effect region is a 2D area specific to a radar, radar site, and wind farm. Similarly to the radar coverages, the adverse effect region should include all 2D locations above which the wind farm affects the radar's performance at the ASRs' surveillance zones. The QUB algorithm has a method for determining an adverse effect region for each radar, radar site, and wind farm triple.

The third assumption addresses the impact of the removal of wind farms in the adverse effect region.

Assumption 3. *Removing a wind farm maintains or increases the 2D fulfillment values of ASRs within the adverse effect regions of the wind farm.*

The rationale behind Assumption 3 is that wind farms impact air surveillance only by interfering with the radar signals, which reduces the capability of radars to surveil targets. Consequently, removing wind farms either eliminates such interference or does not impact air surveillance, resulting in 2D fulfillment values of ASRs being maintained or improved. Figure 9 illustrates Assumption 3 by showing a 2D location marked by a red square, where the removal of a wind farm increases or maintains the 2D fulfillment value of an ASR.



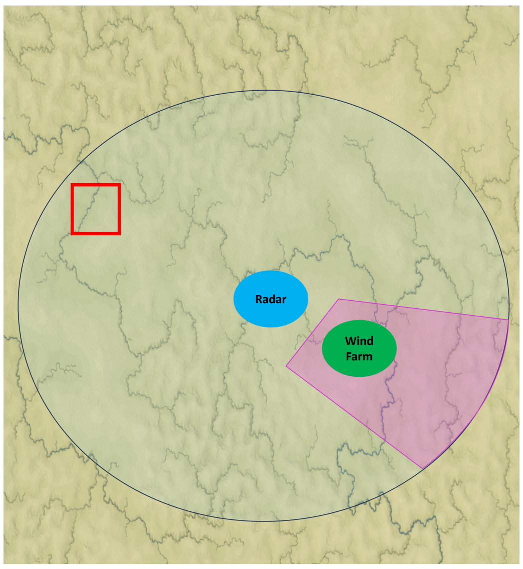
(a) A radar and a wind farm. **(b)** A radar and no wind farm.

Figure 9: The 2D fulfillment value of an ASR at the red square location in Figure 9b is greater than or equal to that at the same location in Figure 9a. The green ellipse depicts a wind farm and the purple-shaded region an adverse effect region. The blue ellipse depicts a radar and the shaded region its coverage.

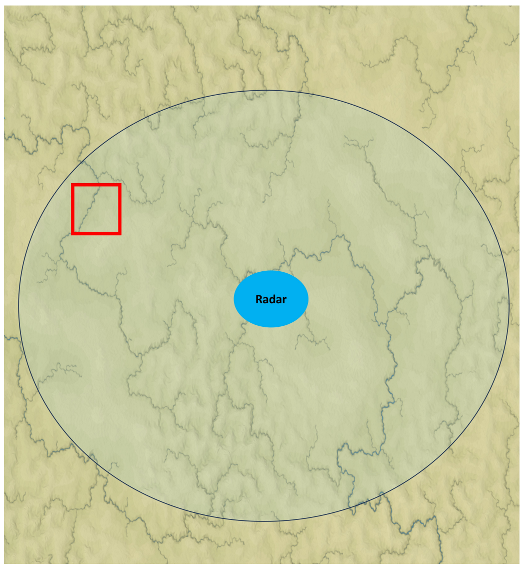
The fourth assumption considers the impact of adding and removing wind farms.

Assumption 4. *Adding or removing a wind farm maintains the 2D fulfillment values of ASRs outside the adverse effect regions of the wind farm.*

The rationale behind Assumption 4 is that if the wind farm being added or removed does not interfere with any radar's ability to detect in a region, then the 2D fulfillment values of ASRs remain unaffected in this region. Figure 10 illustrates Assumption 4 by showing a 2D location, depicted by a red square, where the 2D fulfillment value of an ASR is not impacted by adding or removing a wind farm.



(a) A radar and a wind farm.



(b) A radar and no wind farm.

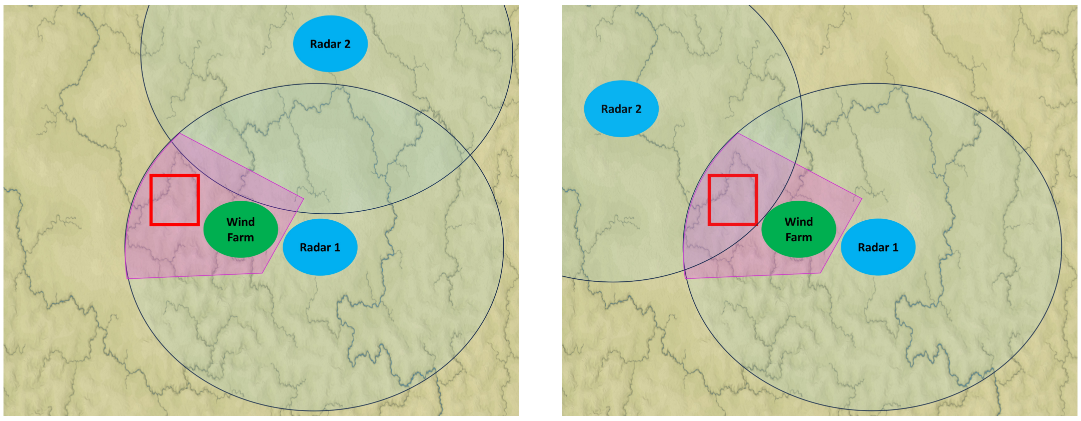
Figure 10: The 2D fulfillment value of an ASR is equal in the 2D location depicted by a red square in Figures 10a and 10b, since the wind farm does not cause any adverse effects in this location. The green ellipse depicts a wind farm, and the purple-shaded region depicts an adverse effect region. The blue ellipse depicts a radar and the shaded region its coverage.

3.2 Computation of the quality upper bound for an alternative

The difference between computing the ASQV and the QUB is that the 2D fulfillment value of an ASR in Equation 3 is replaced with a computationally cheaper upper bound, referred to as the 2D upper bound. Unlike the fulfillment value, this upper bound does not require performance metrics to be evaluated with the computational tool and instead relies on results obtained from previously assessed alternatives. The computation of the 2D upper bound is described next, followed by an explanation of how it is used to determine the QUB for an alternative.

As with the 2D fulfillment value of an ASR, the 2D upper bound is obtained for a combination of an alternative, a 2D location, and an ASR. It is computed using previously determined 2D fulfillment values of the ASR and their corresponding alternatives. To distinguish the alternative for which the 2D upper bound is calculated from those whose 2D fulfillment values were previously determined, the former is referred to as the candidate alternative and the latter as the evaluated alternatives.

The 2D upper bound is computed by iterating over the previously evaluated alternatives. In each iteration, first, the actions to transform the candidate alternative into the evaluated alternative are identified. These actions consist of removing and adding radars and wind farms. To illustrate identifying the actions of transforming the candidate alternative to an evaluated one, consider the candidate alternative in Figure 11a and the evaluated alternative in Figure 11b. The alternatives only differ by the placement of radar 2. Therefore, to transform the candidate alternative in Figure 11a into the evaluated one in Figure 11b, radar 2 is removed from its site in Figure 11a and added to its site in Figure 11b.



(a) The candidate alternative. **(b)** The evaluated alternative.

Figure 11: The candidate alternative and the evaluated alternative differ only by the placement of radar 2. The blue ellipses depict radars, and the shaded regions their coverages. The green ellipse depicts a wind farm, and the purple region its adverse effect region. The red square is the 2D location where the 2D upper bound is computed.

After determining the actions to transform the candidate alternative into the evaluated one, the next step is to assess each action to decide whether or not it maintains or improves the 2D fulfillment value of the ASR. The actions are assessed with Assumptions 1-4 as described in Table 2. If every action preserves or increases the 2D fulfillment value of the ASR, the 2D fulfillment value of the ASR of the previously evaluated alternative is marked as an upper bound candidate. A special case is if the candidate and the evaluated alternatives are the same. Then, no actions are needed to transform the candidate alternative to the evaluated one. In this case, the 2D fulfillment value of the ASR of the evaluated alternative is marked as an upper bound candidate.

Table 2: Determining when actions maintain or improve the 2D fulfillment value of an ASR with Assumptions 1-4.

Action	**When does the action maintain or improve the 2D fulfillment value of an ASR?**
Addition of a radar	According to Assumptions 1 and 2, always.
Removal of a radar	According to Assumption 2, when the radar's coverage does not contain the 2D location of the 2D fulfillment value of the ASR.
Removal of a wind farm	According to Assumptions 3 and 4, always.
Addition of a wind farm	According to Assumption 4, when the adverse effect regions of the wind farm do not contain the 2D location of the 2D fulfillment value of the ASR.

To illustrate marking a 2D fulfillment value as an upper bound candidate, revisit the example of Figure 11. Removing radar 2 from its site in Figure 11a maintains the 2D fulfillment value of the ASR in the 2D location depicted by a red square according to Assumption 2. Adding radar 2 to its site in Figure 11b maintains or increases the 2D fulfillment value of the ASR in the 2D location marked by the red square according to Assumption 1. Since all actions for transforming the candidate alternative in Figure 11a to the evaluated one in Figure 11b maintains or increases the 2D fulfillment value of the ASR, the known 2D fulfillment value of the ASR of the previously evaluated alternative is marked as an upper bound candidate.

The 2D upper bound is the smallest of the upper bound candidates. The smallest upper bound candidate is chosen since it provides the tightest upper bound to the 2D fulfillment value of the ASR. If there are no upper bound candidates, the 2D upper bound is the maximum possible value of the 2D fulfillment value of the ASR. This value is $\sum_{s \in S_j(x,y)} w_{js}$, as presented in Section 2. Algorithm 1 shows the pseudocode for computing a 2D upper bound. The algorithm computes the 2D upper bound for a single location and ASR.

Algorithm 1 Computation of a 2D upper bound

```
   Inputs: Candidate alternative, evaluated alternatives with computed 2D fulfillment
   values of the ASR
1: for each evaluated alternative do
2:     Identify actions to transform the candidate alternative into the evaluated
       alternative
3:     if all actions maintain or increase the 2D fulfillment value of the ASR then
4:         Mark the 2D fulfillment value of the evaluated alternative as an upper
           bound candidate
       end if
   end for
   Return: The smallest upper bound candidate, or the maximum possible value of
   the 2D fulfillment of the ASR if there are no upper bound candidates.
```

To compute the QUB for an alternative, the 2D upper bound is determined for each ASR and 2D location with Algorithm 1. The 2D upper bounds are used to compute upper bounds for the fulfillment of each ASR, similarly to how the fulfillment of an ASR is computed in Equation 4. The upper bound for the fulfillment of an ASR, denoted by $\hat{V}_j(z^i)$, where z^i denotes alternative i and j the ASR, is given by

$$\hat{V}_j(z^i) = \sum_{(x,y) \in S_j^{2D}} \hat{V}_j^{2D}(x, y; z^i), \tag{9}$$

where $\hat{V}_j^{2D}(x, y; z^i)$ is a 2D upper bound for alternative i and ASR j. $(x, y) \in S_j^{2D}$ means all unique (x, y)-coordinate pairs in the 3D-surveillance zone of the ASR. The upper bounds of the fulfillments of ASRs are used to compute the QUB similarly to how the ASQV is computed in Equation 8. Therefore, the QUB, denoted by $\hat{V}(z^i)$ for alternative z^i, is given by

$$\hat{V}(z^i) = \sum_{j=1}^{J} m_j \hat{V}_j(z^i), \tag{10}$$

where m_j are ASR-specific weights representing their importance.

3.3 Inputs, initialization, and iterative phase

The QUB algorithm takes as input a set of alternative radar and wind farm placements and ASRs to identify the alternative that maximizes the ASQV with respect to these ASRs. Initialization of the algorithm involves three tasks. The first is to determine the radar coverages for each feasible radar-site pair, as these are used in the computation of QUBs. The coverage of a radar is computed using the computational tool, which determines the minimum altitude at each 2D location where targets are within line of sight of the radar, accounting for terrain obstructions. The coverage of a radar consists of all the 2D locations, for which this altitude is lower than the maximum altitude within the ASRs' surveillance zones.

The second of the three tasks is determining an adverse effect region for each radar, radar site, and wind farm triple. As the radar coverages, these regions are essential for computing QUBs. The adverse effect region for a radar, radar site, and wind farm triple is established by first determining whether there is a direct line of sight from the radar to the wind farm. If no line of sight exists, the wind farm cannot produce adverse effects on the radar, and the adverse effect region is therefore empty. If a line of sight is present, the adverse effect region consists of all 2D locations that satisfy the following three conditions.

1. The 2D location is part of the coverage of the radar.
2. Consider a polar coordinate system centered at the radar. The absolute difference between the azimuth coordinate of the 2D location and any wind turbine in the wind farm is less than 8 degrees.
3. The distance between the radar and the 2D location is greater than the distance between the radar and the wind turbine closest to the radar minus 20km.

Figure 12 illustrates an adverse effect region by showing the relevant angles and distance on which the region is based.

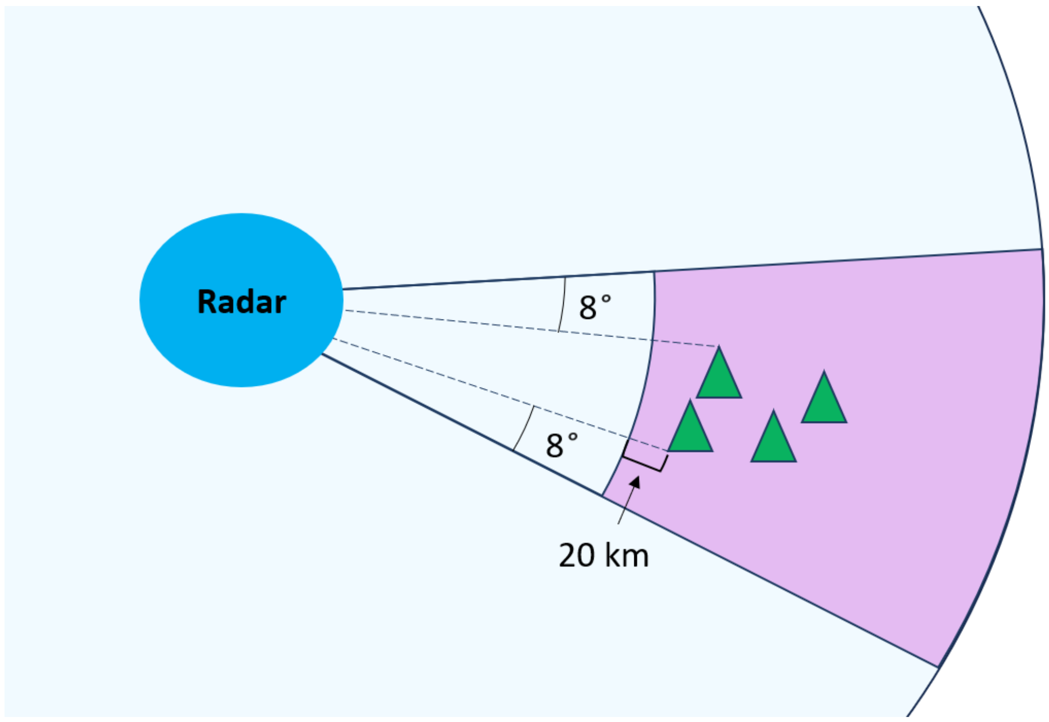


Figure 12: The adverse effect region is depicted in purple. The green triangles are wind turbines, the radar is a blue ellipse, and the coverage of the radar is depicted in light blue.

The third and final task for initializing the algorithm is to compute the 2D fulfillment values of ASRs with the computational tool and Equation 3 for two initial radar and wind farm placement combinations. These combinations are not actual alternatives, since they are not feasible. The two combinations consist of

1. Placing a radar at all radar sites and a wind farm at all wind farm sites.
2. Placing a radar at all radar sites and no wind farms at any site.

The computed 2D fulfillment values of ASRs are stored to be used by Algorithm 1 for computing 2D upper bounds. These radar and wind farm placement combinations are chosen since they determine the maximum 2D fulfillment values of ASRs with and without wind farms.

After the initialization, the algorithm enters the iterative phase. The iterative phase starts by computing QUBs for all alternatives separately with Algorithm 1 and Equations 9 and 10. Then, an exhaustive search is used to identify the maximum QUB of the alternatives and the corresponding alternative. The maximum computed ASQV is initialized to the value 0, since the ASQV of no alternative has been computed yet.

The algorithm iterates while the maximum QUB is greater than the maximum computed ASQV. Each iteration starts by computing the ASQV for the alternative with the maximum QUB. When the ASQV is computed, the 2D fulfillment values of ASRs determined with Equation 3 are stored to be used with Algorithm 1 for calculating QUBs. Thereby, computing the ASQV of an alternative impacts the QUBs. Therefore, after computing the ASQV of the alternative with the maximum QUB, the QUB of every alternative is recomputed with Algorithm 1 and Equations 9 and 10. Each iteration ends with determining the maximum QUB with an exhaustive search and determining the maximum computed ASQV. Algorithm 2 shows the pseudocode of the QUB algorithm.

Algorithm 2 The quality upper bound algorithm

```
    Inputs: Set of all alternatives, ASRs
    Initialization:
 1:     Determine radar coverages
 2:     Determine adverse effect regions
 3:     Compute the 2D fulfillment values of ASRs of two initial radar and wind
        farm placement combinations
    Iterative phase:
 4:     Compute the QUB of every alternative
 5:     Determine the maximum QUB and the alternative corresponding to it with
        an exhaustive search
 6:     Initialize the maximum computed ASQV with 0
 7:     while Maximum QUB > Maximum computed ASQV do
 8:         Compute the ASQV of the alternative with the maximum QUB
 9:         Compute the QUB of every alternative
10:         Determine the maximum QUB and the alternative corresponding to it
            with an exhaustive search
11:         Determine the maximum computed ASQV
        end while
    Return: Optimal alternative
```

3.4 From assumptions to optimality

The QUB algorithm (Algorithm 2) finds the alternative with the maximum ASQV if the assumptions presented in Section 3.1 hold. This is established by first showing that the algorithm terminates in finite time. Then, it is shown that, upon termination, the algorithm has found the optimal alternative under the assumptions.

3.4.1 On finite time termination

Algorithm 2 terminates since all individual steps of the algorithm and the loop starting at Step 7 terminate. Steps 1-3, 6, 8, and 11 of Algorithm 2 terminate, since they consist of finite sequences of operations. Steps 4, 5, 9, and 10 require enumerating all alternatives, which can be carried out given enough time, since the number of alternatives is finite. Steps 4 and 9 involve computing QUBs with Algorithm 1 and Equations 9 and 10, which consist of a finite number of operations. Thus, all individual steps of Algorithm 2 terminate.

The loop at Step 7 terminates in finite time because the number of alternatives is finite, and the loop computes the ASQV of each alternative at most once. This follows from the fact that the loop always selects the alternative with the maximum QUB for ASQV computation, and once an alternative's ASQV has been computed, its QUB cannot exceed its ASQV. Consequently, the same alternative cannot be selected again for ASQV computation, as this would cause the loop to terminate. Thus, the loop terminates after a finite number of iterations.

3.4.2 On optimality

When Algorithm 2 has terminated, the loop starting at step 7 has completed. At this point, there is no alternative whose QUB is greater than the highest computed ASQV. Under the assumptions outlined in Section 3.1, the QUB of any alternative is greater than or equal to its ASQV. Therefore, once the algorithm stops, it has identified the optimal alternative, provided that the assumptions hold. It is next shown that, under the given assumptions, the QUB is greater than the ASQV.

When the assumptions hold, the 2D upper bounds are greater than or equal to 2D fulfillment values of ASRs. This is because Algorithm 1 chooses the 2D upper bound from upper bounds candidates, which are greater than or equal to their respective 2D fulfillment values under the assumptions. Consequently, the QUB of an alternative is greater than or equal to its ASQV, since

$$\begin{aligned} V(z^i) &= \sum_{j=1}^{J} m_j \sum_{(x,y)\in S_j} V_j^{2D}(x, y; z^i) \\ &\leq \sum_{j=1}^{J} m_j \sum_{(x,y)\in S_j} \hat{V}_j^{2D}(x, y; z^i) \\ &= \hat{V}(z^i). \end{aligned}$$

3.5 Implications of assumption violations

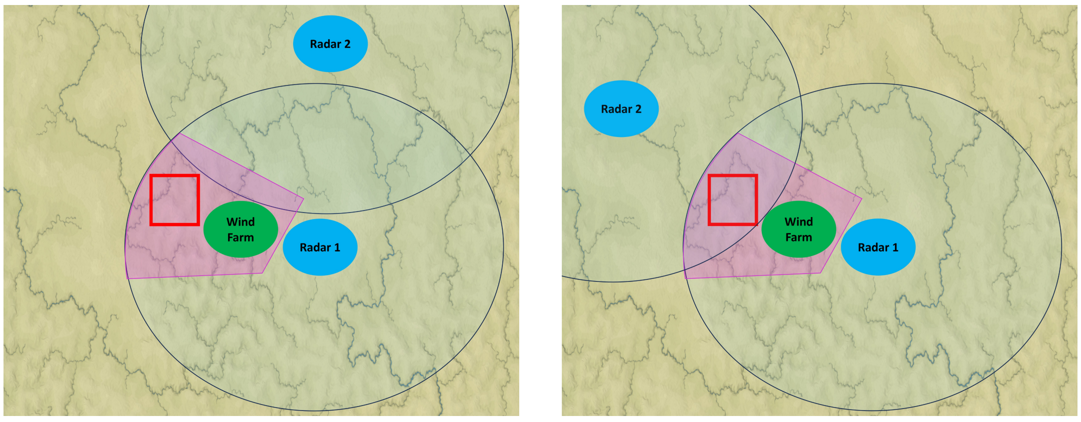
If the assumptions outlined in Section 3.1 are violated, the QUB algorithm is not guaranteed to identify the optimal alternative, as the QUB of an alternative may be smaller than its ASQV. Next, it is illustrated through an example how violating the assumptions can cause the QUB to be smaller than the ASQV.

The 2D upper bound for the alternative shown in Figure 13a is computed using Algorithm 1 at the location marked by a red square. This alternative is referred to as the candidate alternative. The alternative in Figure 13b is referred to as the evaluated alternative. Its 2D fulfillment value of the ASR has been determined with the computational tool and Equation 3 at the 2D location depicted by the red square.

At Step 2 of Algorithm 1, the actions for transforming the candidate alternative into the evaluated one are identified. These actions consist of removing radar 2 from its site in Figure 13a and adding radar 2 to its site in Figure 13b. Step 3 of Algorithm

1 analyzes whether all the actions maintain or increase the 2D fulfillment value of the ASR at the location illustrated by the red square, which they clearly do under the assumptions. Thus, the evaluated alternative's 2D fulfillment value is marked as an upper bound candidate, and since it is the only upper bound candidate, it is selected as the 2D upper bound.

If the Assumptions 1 and 2 are violated, the 2D upper bound in the above example may be smaller than the 2D fulfillment value of the ASR for the candidate alternative, if it were computed. There are multiple ways in which the assumptions could be violated. For instance, if the coverage of radar 2 in Figure 13a is too small and in reality the radar impacts the 2D fulfillment value of an ASR at the red square, removing the radar could diminish the 2D fulfillment value of the ASR. Furthermore, if the computation of the 2D fulfillment value is stochastic, the 2D upper bound could be smaller than the 2D fulfillment value by chance. When 2D upper bounds are smaller than the 2D fulfillment values, the QUB of an alternative may be smaller than its ASQV.



(a) The candidate alternative. **(b)** The evaluated alternative.

Figure 13: A candidate alternative and an evaluated alternative. The blue ellipses depict radars, and the shaded regions their coverages. The green ellipse depicts a wind farm, and the purple region its adverse effect region. The red square is the 2D location where the 2D upper bound is computed.

3.6 Limitations

The QUB algorithm has two main limitation. First, the QUB algorithm's guarantee of optimality depends on the assumptions outlined in Section 3.1 being satisfied. Secondly, the QUB algorithm enumerates all alternatives in Steps 4, 5, 9, and 10. As discussed in Section 2, the number of alternatives increases rapdily with the number of feasible sites for radars and wind farms. The QUB algorithm enumerates all alternatives iteratively, specifically in Steps 9 and 10 of the while loop that begins at Step 7.

To illustrate the issue of iteratively performing Step 9, suppose that there are 10^6 alternatives and the computation of a QUB for a single alternative with Algorithm 1 and Equations 10 and 9 takes 10^{-2} seconds. Then, the computation of the QUB for all alternatives in Step 9 takes 10^4 seconds, i.e., 2h 47min. As computing the ASQV of an alternative with the computational tool and Equations 3, 4, and 8 takes minutes, Step 9 of becomes a bottleneck. The next section presents the extended QUB algorithm to address the issue of iteratively enumareting all alternatives.

4 The extended quality upper bound algorithm

This section presents the extended QUB algorithm, a variant of the QUB algorithm introduced in Section 3. The extended QUB algorithm differs from the QUB algorithm in using more efficient search strategies. As discussed in Section 3.6, in Steps 4, 5, 9, and 10 of the QUB algorithm (Algorithm 2), computing the QUB of every alternative and selecting the alternative with the maximal QUB with an exhaustive search scales poorly, as the number of alternatives increases rapidly with the number of feasible sites for radars and wind farms. The extended QUB algorithm overcomes this challenge by replacing these steps with a random restart hill climbing (RRHC) algorithm. RRHC finds an alternative with a near-maximal QUB without computing the QUB of every alternative by repeatedly calling a hill climbing search.

However, replacing Steps 4, 5, 9, and 10 of Algorithm 2 with RRHC results in an algorithm that is not guaranteed to find an alternative whose ASQV is greater than or equal to the QUB of every other alternative. Consequently, the algorithm does not necessarily return an optimal alternative under the assumptions presented in Section 3.1. To ensure optimality under the assumptions, the extended QUB algorithm consists of two phases: exploration and refinement. During the exploration phase, RRHC is used to identify an alternative with a high ASQV, although this alternative is not guaranteed to be optimal. In the refinement phase, this alternative is improved until its ASQV exceeds the QUB of all others, resulting in an optimal solution under the assumptions.

Confirming that an alternative's ASQV is greater than or equal to the QUB of all others in the refinement phase requires computing the QUB of every alternative, which can be time-consuming. For example, evaluating the QUB of 10^5 alternatives takes approximately 17 minutes when the QUB of a single alternative requires 10^{-2} seconds. Furthermore, doing so for 10^{14} alternatives would take around $30,000$ years.

Due to this computational cost, the refinement phase computes the QUB of every alternative only when the total number of alternatives is less than 10^5. This threshold was selected based on empirical observations from the numerical experiments in Section 5, which indicate that, beyond 10^5 alternatives, the computational cost of verifying optimality outweighs the potential improvement in solution quality. When the number is larger, a local optimization algorithm is used instead. This algorithm returns an alternative whose ASQV is greater than or equal to the QUBs of its neighbors. Neighbors are defined as alternatives obtained by relocating a single radar or wind farm.

This section is structured as follows. First, the inputs, initialization, exploration phase, and refinement phase of the extended QUB algorithm are detailed. Next, the RRHC in the exploration phase is presented. Then, the local optimization algorithm in the refinement phase is introduced. Moreover, the optimality of the alternative returned by the extended QUB under the assumptions of Section 3.1 is established. Finally, the algorithm's limitations are discussed, and the QUB and extended QUB algorithms are compared.

4.1 Inputs, initialization, exploration phase, and refinement phase

The inputs of the extended QUB algorithm are the same as those of the QUB algorithm: the set of all alternatives and ASRs. The extended QUB algorithm is initialized identically to the QUB algorithm (see Section 3.3) by determining radar coverages and adverse effect regions and computing the 2D fulfillment values of ASRs of two initial radar and wind farm placement combinations.

After initialization, the algorithm enters the exploration phase. This phase begins by initializing the maximum computed ASQV to 0, since the ASQV of no alternative has yet been computed. RRHC is then employed to find an alternative with a near-maximal QUB. The algorithm then enters a loop that continues as long as the QUB of the alternative found by RRHC exceeds the maximum ASQV.

Each iteration of the loop consists of three steps. First, the ASQV for the alternative found by RRHC is determined using the computational tool and Equations 3, 4, and 8. When the ASQV is computed, the 2D fulfillment values of ASRs are stored for computing QUBs. Next, the maximum ASQV is updated by selecting the highest ASQV among the computed ASQVs. Finally, RRHC is utilized to find an alternative with a QUB exceeding the current maximum ASQV. Once the QUB of the alternative found by RRHC is no longer greater than the maximum ASQV, the loop terminates and the exploration phase ends.

When the exploration phase concludes, the algorithm transitions into the refinement phase. All radar coverages, adverse effect regions, computed ASQVs, and 2D fulfillment values of ASRs are carried over. The maximum computed ASQV is likewise retained. The behavior of the refinement phase depends on the number of alternatives.

If the number of alternatives is smaller than 10^5, the refinement phase begins by computing the QUB for every alternative. The alternative with the highest QUB is then identified through exhaustive search. From this point, the algorithm enters an iterative process that continues as long as the maximum QUB exceeds the maximum computed ASQV.

The following steps are repeated in the iterative process. The ASQV of the alternative with the maximum QUB is calculated, and the maximum ASQV is updated based on this result. Following this, all alternatives with a QUB greater than the current maximum ASQV are identified, and their QUBs are computed. The highest among these is set as the new maximum QUB. The process terminates once no alternative has a QUB greater than the maximum ASQV.

If the number of alternatives is greater than 10^5, the refinement phase applies the local optimization algorithm, starting from the alternative with the maximum computed ASQV. The algorithm identifies an alternative whose ASQV exceeds the QUBs of its neighbors. Section 4.3 presents the local optimization algorithm in detail.

Once the refinement phase is complete, the extended QUB algorithm returns the alternative with the highest computed ASQV. Algorithm 3 provides the complete pseudocode of the extended QUB algorithm.

Algorithm 3 The extended quality upper bound algorithm

```
    Inputs: Set of all alternatives, ASRs
    Initialization:
 1:     Determine radar coverages
 2:     Determine the adverse effect regions
 3:     Compute the 2D fulfillment values of ASRs of two initial radar and wind
        farm placement combinations
    Exploration phase:
 4:     Initialize the maximum computed ASQV of an alternative with 0
 5:     Determine the maximum QUB with RRHC
 6:     while Maximum QUB > Maximum computed ASQV do
 7:         Compute the ASQV of the alternative with the maximum QUB
 8:         Determine the maximum computed ASQV
 9:         Determine the maximum QUB with RRHC
        end while
    Refinment phase:
10:     if The number of alternatives is less than $10^5$ then
11:         Compute the QUB of every alternative
12:         Determine the maximum QUB with an exhaustive search
13:         Determine the maximum computed ASQV
14:         while Maximum QUB > Maximum computed ASQV do
15:             Compute the ASQV of the alternative with the maximum QUB
16:             Determine the maximum computed ASQV
17:             Identify all alternatives with a QUB exceeding the maximum computed
                ASQV
18:             Compute the QUBs of these alternatives
19:             Set the maximum QUB to the highest QUB computed in the last step
            end while
20:     else if The number of alternatives is larger than or equal to $10^5$ then
21:         Employ the local optimization algorithm starting from the alternative with
            the maximum computed ASQV
        end if
    Return: The alternative with the maximum computed ASQV and its ASQV
```

4.2 Random restart hill climbing algorithm in the exploration phase

The RRHC algorithm takes as input the set of alternatives, a maximum number of restarts, and the maximum computed ASQV, determined in Steps 4 and 8 of Algorithm 3. Its objective is to identify an alternative with a maximal QUB, although it is not guaranteed to do so. The algorithm continues restarting a hill climbing search from random alternatives until either the search returns an alternative with a QUB exceeding the maximum computed ASQV or the maximum number of restarts is reached. The maximum number of restarts used in this thesis is 10. The pseudocode for RRHC is provided in Algorithm 4. The hill climbing search is presented next.

Algorithm 4 Random restart hill climbing

```
   Inputs: Set of all alternatives, maximum number of restarts, the maximum
   computed ASQV
1: Set the number of restarts and the maximum QUB to zero
2: while Number of restarts $<$ Maximum number of restarts and
   Maximum QUB $\leq$ Maximum computed ASQV do
3:     Select a random alternative
4:     Apply a hill climbing search starting from the random alternative
5:     Set the result of the hill climbing search as the maximum QUB
6:     Increment the number of restarts
   end while
   Return: The maximum QUB and the corresponding alternative
```

The hill climbing search in Step 6 of Algorithm 3 takes an initial alternative as input and begins by setting it as the current alternative. The neighbors of the current alternative are all the alternatives obtained by relocating a single radar or wind farm. The hill climbing search iterates while the current solution has a neighbor with a higher QUB. In each iteration, a list of all neighbors of the current alternative is created. A random neighbor is repeatedly selected from the list of neighbors, and its QUB is computed. If this QUB exceeds that of the current alternative, the selected neighbor replaces it as the new current alternative, and the search continues with the neighbors of this new alternative. If no neighbor yields a higher QUB, the algorithm terminates and returns the current alternative along with its QUB. Algorithm 5 shows the pseudocode of the hill climbing search.

Algorithm 5 Hill climbing search

```
    Inputs: Initial alternative
 1: Set the initial alternative as the current alternative
 2: Compute the QUB of the current alternative
 3: Create a list of the neighbors of the current alternative
 4: while The list of neighbors is not empty do
 5:     Select a random neighbor from the list of neighbors
 6:     Remove the selected neighbor from the list
 7:     Compute the QUB of the selected neighbor
 8:     while QUB of the selected neighbor $\leq$ QUB of the current alternative and
        the list of neighbors is not empty do
 9:         Select a random neighbor from the list of neighbors
10:         Remove the selected neighbor from the list
11:         Compute the QUB of the selected neighbor
        end while
12:     if QUB of the selected neighbor > QUB of the current alternative then
13:         Set the selected neighbor as the current alternative
14:         Create a list of the neighbors of the current alternative
        end if
    end while
    Return The current alternative and its QUB
```

4.3 Local optimization algorithm in the refinement phase

The local optimization algorithm, invoked in Step 21 of Algorithm 3, receives an initial alternative along with its computed ASQV. It searches for an alternative whose ASQV exceeds the QUB of all neighboring alternatives. The algorithm begins by setting the initial alternative as the current alternative.

In each iteration, the QUBs of the neighboring alternatives are determined. If no neighbor has a QUB greater than the ASQV of the current solution, the algorithm terminates and returns this solution along with its ASQV. Otherwise, the neighbor with the highest QUB is selected, and its ASQV is computed. If this ASQV exceeds that of the current solution, the neighbor becomes the new current solution. The iteration continues until no further improvement is found. Algorithm 6 shows the pseudocode of the local optimization algorithm.

Algorithm 6 Local optimization algorithm

```
   Inputs: Initial alternative and its computed ASQV
1: Set the initial alternative as the current alternative
2: Compute the QUBs of all the neighbors of the current alternative
3: Determine the maximum QUB of the neighbors
4: while Maximum QUB of the neighbors > ASQV of the current alternative do
5:     Compute the ASQV of the neighbor with the maximum QUB
6:     if ASQV of the neighbor with the maximum QUB > ASQV of the current
       alternative
7:         Set the neighbor with the maximum QUB as the current alternative
       end if
8:     Compute the QUBs of all the neighbors of the current alternative
9:     Determine the maximum QUB of the neighbors
   end while
   Return: The current alternative and its computed ASQV
```

4.4 From assumptions to optimality

The extended QUB algorithm returns an alternative with the maximal ASQV if the assumptions in Section 3.1 hold and the number of alternatives is less than 10^5. If the number of alternatives is at least 10^5, it returns an alternative whose ASQV is greater than or equal to that of any neighbor. These results are established by first showing that the algorithm terminates in finite time. Under the assumptions, it is shown that the algorithm returns an alternative with the maximal ASQV when there are fewer than 10^5 alternatives, and an alternative whose ASQV is at least as high as that of any neighbor when the number is 10^5 or more.

4.4.1 On finite time termination

Finite time termination of the extended QUB algorithm is nontrivial due to the presence of several iterative components. These components include the loops in Algorithms 3, 4, 5, and 6. Specifically, the loops begin at Step 6 and Step 14 in Algorithm 3, at Step 2 in Algorithm 4, at Steps 4 and 8 in Algorithm 5, and at Step 4 in Algorithm 6.

The loops starting at Steps 6 and 14 of Algorithm 3 and at Step 4 of Algorithm 6 terminate by a similar argument as the one presented in Section 3.4.1 for the loop of the QUB algorithm. The argument is that if any of these loops were to continue indefinitely, they would compute an alternative's ASQV more than once. However, each loop would halt before computing the alternative's ASQV a second time.

The loop starting at Step 2 of Algorithm 4 stops after the maximum number of restarts given as input to the algorithm. The loop starting at Step 8 of Algorithm 5 repeatedly removes neighbors from the list and necessarily ends once all neighbors have been removed. Algorithm 5 also terminates, since each iteration replaces the current alternative with one that has a strictly higher QUB, and the number of alternatives is finite. As a result, such replacements can occur only a finite number of times. Thus, all the iterative components of the extended QUB algorithm terminate in finite time.

4.4.2 On optimality

When the number of alternatives is smaller than 10^5, the extended QUB algorithm terminates when the while loop starting at Step 14 stops. This occurs when the algorithm has found an alternative whose ASQV is greater than the QUB of every other alternative. As seen in Section 3.4.2, the QUB of an alternative is greater than its ASQV if the assumptions of Section 3.1 hold. Thus, under the assumptions, the extended QUB algorithm returns an alternative with maximal ASQV.

For 10^5 or more alternatives, the extended QUB algorithm terminates when the while loop starting at Step 4 of Algorithm 6 stops. The while loop stops when the algorithm has found an alternative whose computed ASQV exceeds the QUB of all its neighbors. Thus, under the assumptions, the extended QUB algorithm finds an alternative whose ASQV is greater than or equal to its neighbors' ASQVs.

4.5 Limitations

As discussed in Section 3.5, violating the assumptions of Section 3.1 may cause the QUBs of alternatives to be smaller than their ASQVs. Consequently, when the computational tool violates the assumptions, the extended QUB algorithm is not guaranteed to return an alternative with the maximal ASQV when the number of alternatives is smaller than 10^5, nor an alternative whose ASQV is greater than or equal to its neighbors' ASQVs when the number of alternatives is at least 10^5.

4.6 Comparison of the QUB algorithms

Under the assumptions, the extended QUB Algorithm identifies an optimal alternative in problems with fewer than 10^5 alternatives. In problems with more alternatives, it returns an alternative whose ASQV exceeds that of all the alternatives differing by a single radar or wind farm placement. While this alternative can be an optimal alternative, it is not guaranteed to be.

The QUB algorithm always returns the optimal alternative when the assumptions hold. However, it is slower than the extended algorithm, regardless of the number of alternatives.

5 Numerical experiments

This section demonstrates the effectiveness of the QUB algorithms through numerical experiments. The algorithms are applied to 11 example problems with varying sizes. The experiments measure each algorithm's execution time, the number of ASQV computations, and the quality of the alternatives they return. Quality is examined by comparing the algorithms' outputs to one another and the known maximum ASQVs in small problems, where the ASQV of every alternative can be computed within a few hours. The ASQV is computed using an existing computational tool (see, Lahti, 2022; Virtanen, 2024).

The assumptions presented in Section 3.1 are violated in the numerical experiments. Therefore, the optimality of the returned alternatives discussed in Sections 3.4 and 4.4 is not guaranteed. The impact of the assumption violations on the quality of the algorithms' outputs is assessed. In the following, the example problems are first introduced, and then the results are presented.

5.1 Example problems

The example problems include five small problems, five medium problems, and one large problem. The objective of each problem is to place radars and wind farms to fulfill the same 18 ASRs. The problems are introduced by first outlining the decision alternatives, followed by descriptions of the ASRs.

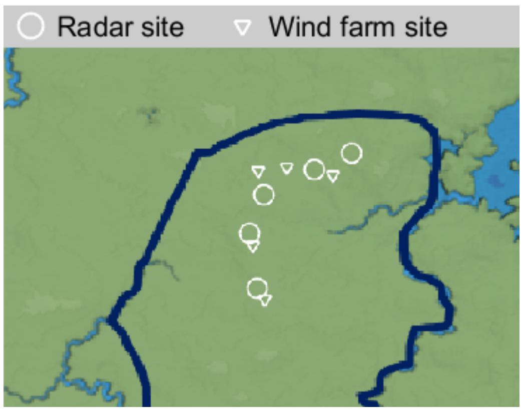
5.1.1 Decision alternatives

Each small problem includes five feasible radar sites and five feasible wind farm sites. The task is to place three radars and three wind farms at these sites. The total number of alternatives per problem is 100. Figure 14 illustrates the feasible sites for the small problems.

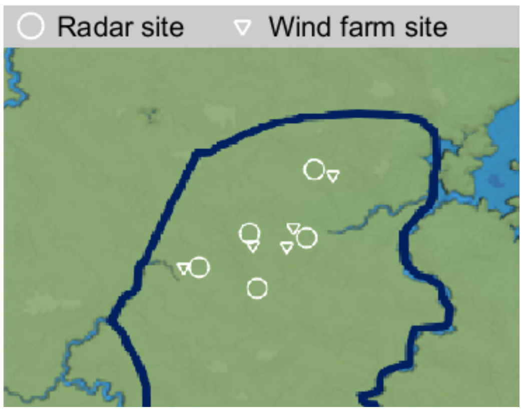
In the medium problems, there are 10 radar sites and 11 wind farm sites. The objective is to place five radars and five wind farms. Each medium problem has approximately 10^5 alternatives. Figure 15 shows the feasible sites.

The large problem contains 21 radar sites and 35 wind farm sites, with the goal of placing 10 radars and 20 wind farms. This results in around 10^{15} alternatives. Figure 16 displays the feasible sites for this problem.

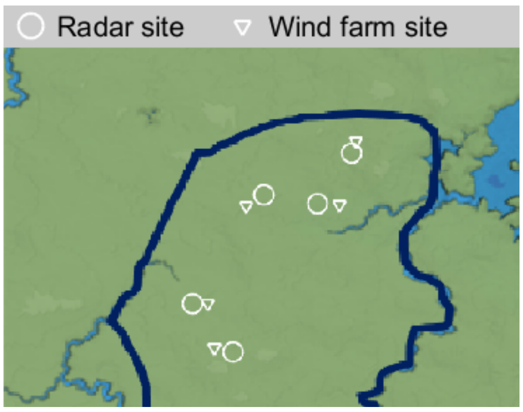
The radars considered in all problems are ground-based, monostatic, and have a range of 300 kilometers. The wind farms consist of 50 wind turbines arranged in a rectangular formation, spaced 600 meters apart. Each turbine features a tower that is 210 meters high and 8 meters wide. The blades are 90 meters long, resulting in a total structure height of 300 meters when a blade is in the vertical position.



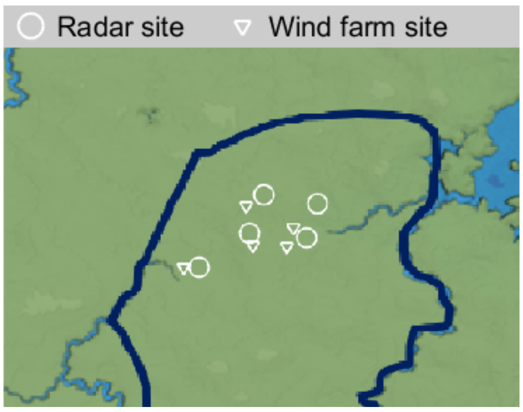
(a) Small problem number 1



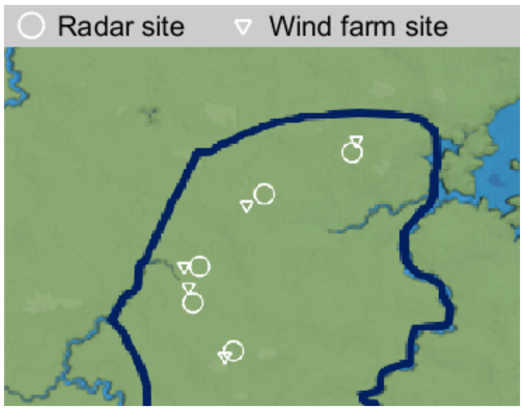
(b) Small problem number 2.



(c) Small problem number 3.

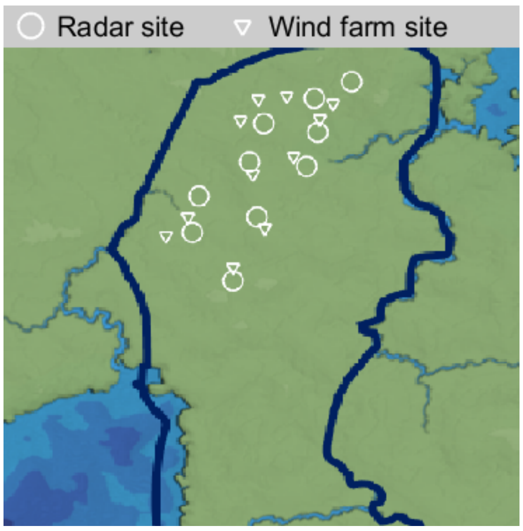


(d) Small problem number 4.

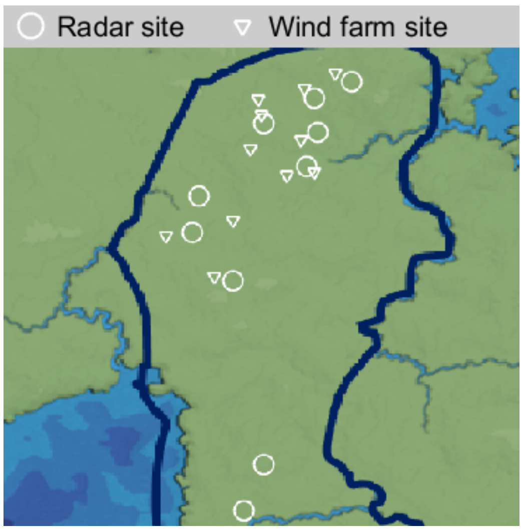


(e) Small problem number 5.

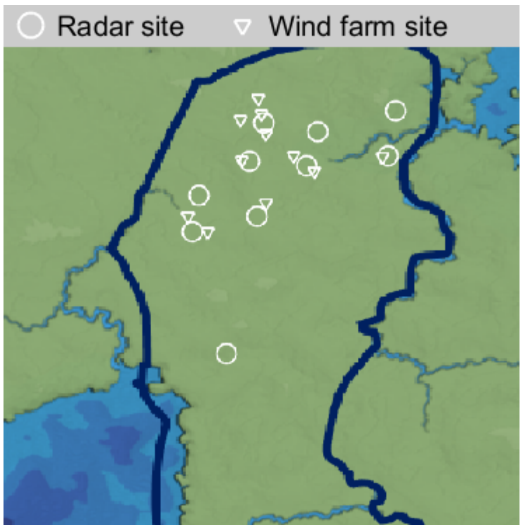
Figure 14: The feasible radar and wind farm sites of the five small problems.



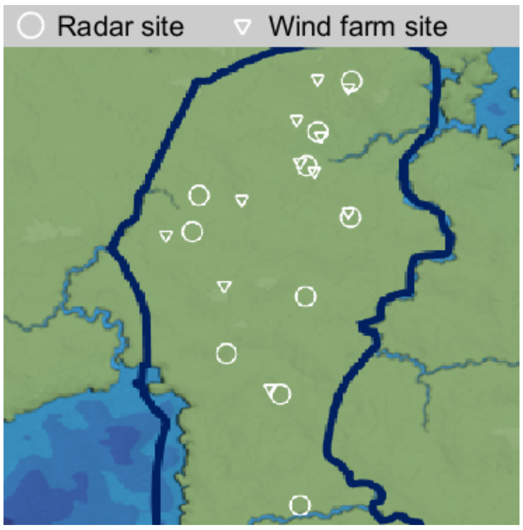
(a) Medium problem number 1



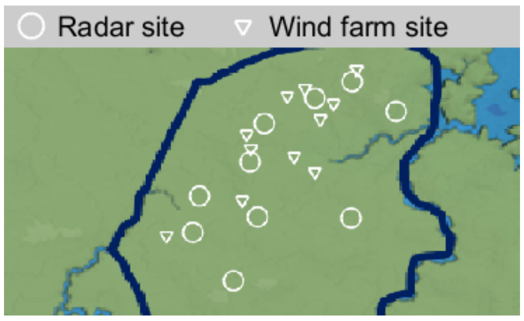
(b) Medium problem number 2.



(c) Medium problem number 3.



(d) Medium problem number 4.



(e) Medium problem number 5.

Figure 15: The feasible radar and wind farm sites of the five medium problems.

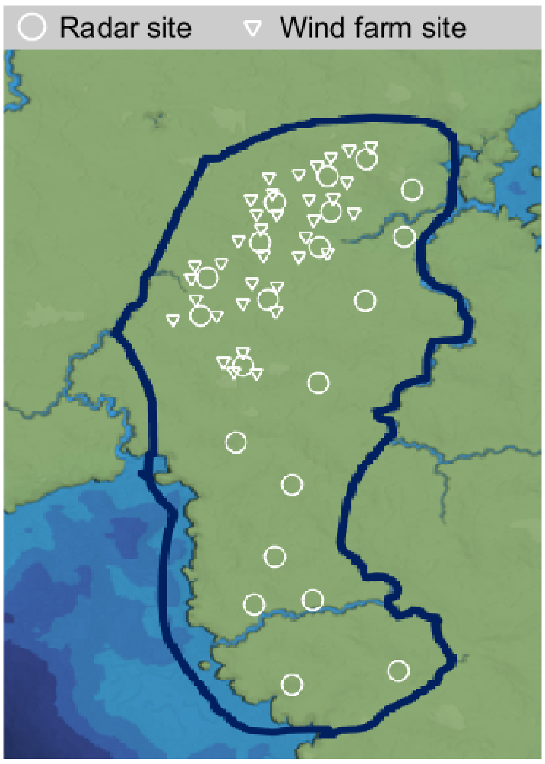


Figure 16: The feasible radar and wind farm sites of the large problem.

5.1.2 Air surveillance requirements

The objective of all small, medium, and large problems is to maximize the fulfillment of the same 18 ASRs. As presented in Section 2, an ASR consists of a surveillance zone specified by a 2D zone and an altitude interval, a target type, a quality statement, and a priority.

The target of all 18 ASRs is a cruise missile. Each of the 18 ASRs has a unique surveillance zone with no overlap. Figure 17 depicts the 2D zones of the ASRs with red polygons. The altitude interval is 1-3km for each ASR.

Each ASR has one of the four priorities: P1, P2, P3, and P4. Figure 18 illustrates these priorities. Table 3 shows the weights corresponding to each priority. The weights were computed with the centroid weights method based on the priorities (see, e.g., Ahn, 2011).

There are four quality statements denoted by Q1, Q2, Q3, and Q4. Table 4 details these quality statements. Figure 19 shows the quality statement of each requirement. The linear consequence value functions are used for the quality statements according to Equations 1 and 2.

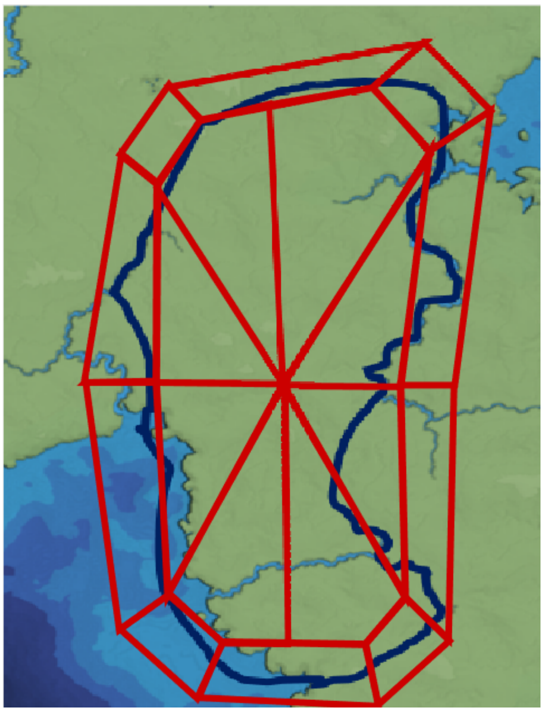


Figure 17: The 2D zones of the ASRs depicted by red polygons.

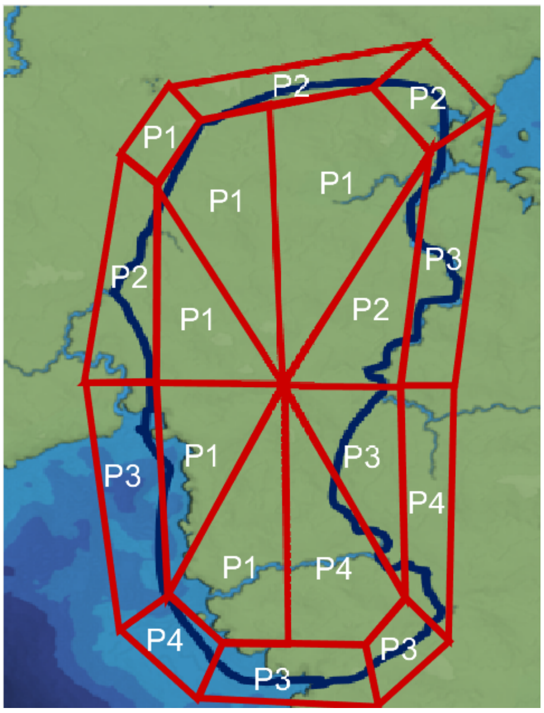


Figure 18: The priorities of the ASRs depicted by text within the 2D zones.

Table 3: The weights corresponding to the priorities

Priority	Weight
P1	0.1136
P2	0.0471
P3	0.0221
P4	0.0064

Table 4: The best and worst values for each performance metric given in the quality statement

Name of quality statement	Probability of detection		Time between observations (s)		Track accuracy (m)	
	Worst	Best	Worst	Best	Worst	Best
Q1	0.1	0.5	45	30	3000	1000
Q2	0.1	0.7	45	10	3000	200
Q3	0.1	0.8	45	6	3000	100
Q4	0.1	0.9	45	3	3000	50

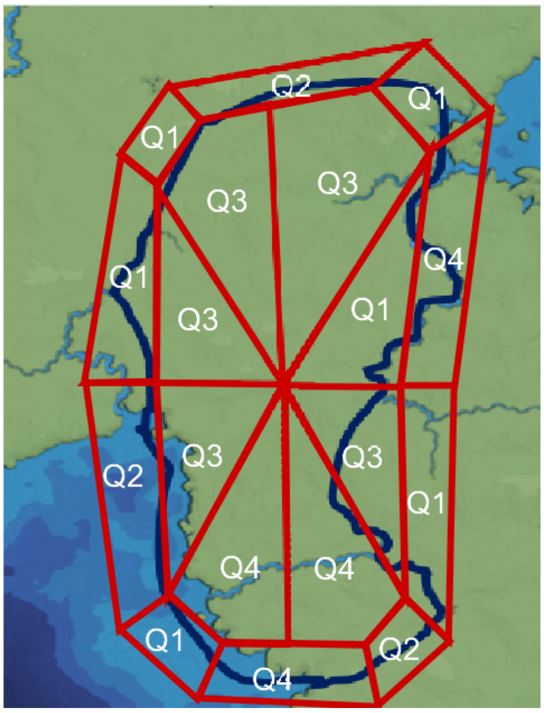


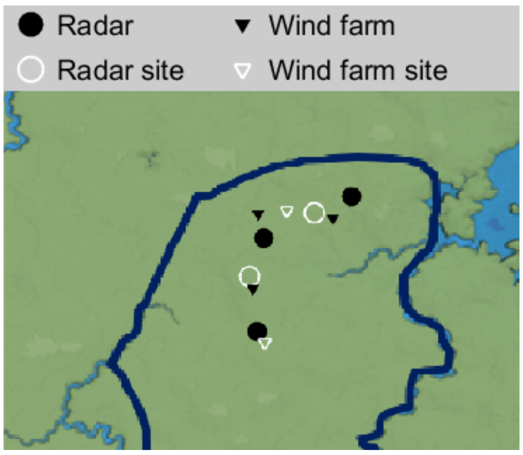
Figure 19: The quality statements of the ASRs depicted by text within the 2D zones.

5.2 Results

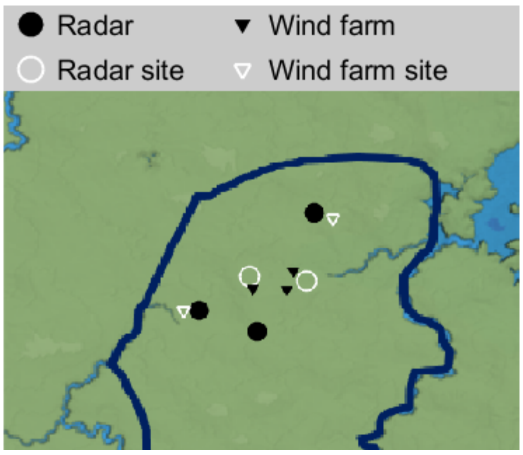
The results of the numerical experiments are presented for the small, medium, and large problems. Moreover, the impact of assumption violations is discussed. The experiments were run on a laptop with an Intel Core i5-6300U 2.40GHz CPU and 8GB RAM.

5.2.1 Small problems

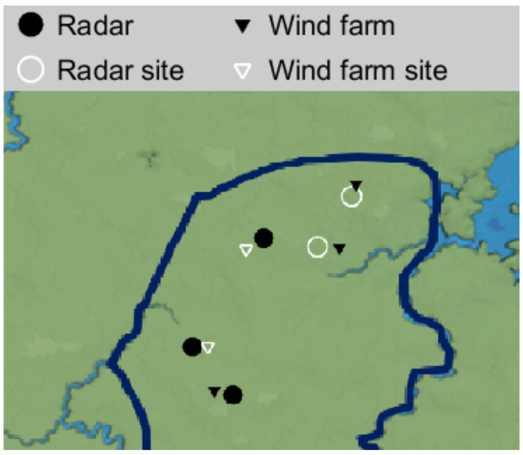
The QUB algorithm was applied to the small problems. Figure 20 shows the alternatives that the algorithm returned. These alternatives were confirmed optimal by computing the ASQV of all 100 alternatives in each small problem. Computing the ASQV of all alternatives took 4 hours. The QUB algorithm terminated on average in 12 computed ASQVs and 44 minutes. The number of computed ASQVs includes the two initial radar and wind farm combinations at Step 3 of Algorithm 2.



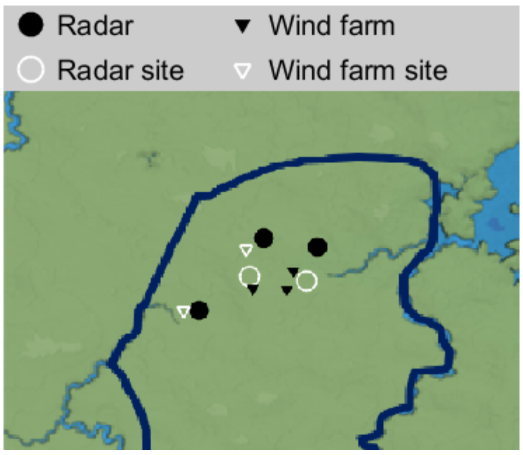
(a) Small problem number 1



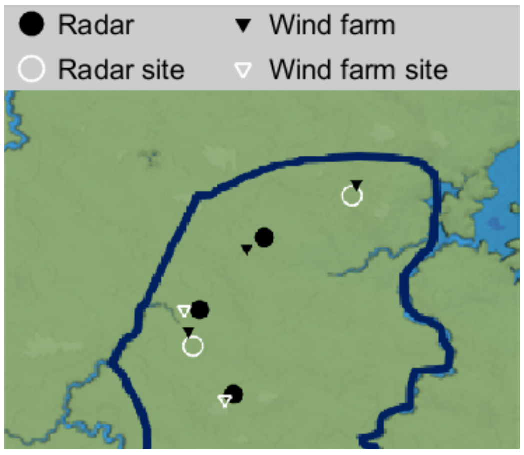
(b) Small problem number 2.



(c) Small problem number 3.



(d) Small problem number 4.



(e) Small problem number 5.

Figure 20: The optimal radar and wind farm placements in the five small problems.

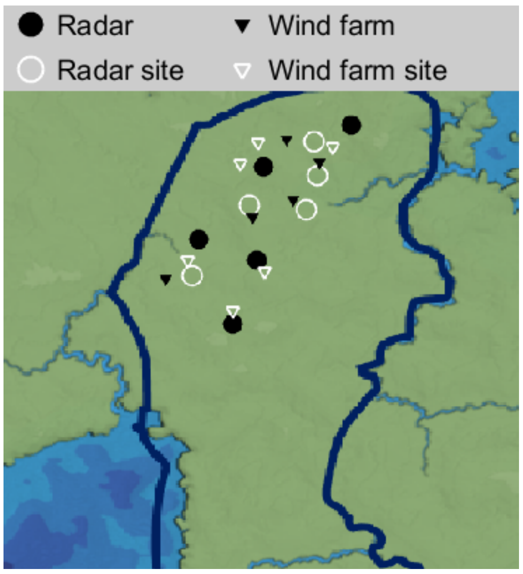
5.2.2 Medium problems

Computing the ASQV of all alternatives in any of the medium problems would take approximately one year. In contrast, the QUB algorithm returned an alternative in 17 hours by computing 20 ASQVs in medium problem 1. Of the 17 hours, 15 were spent computing QUBs.

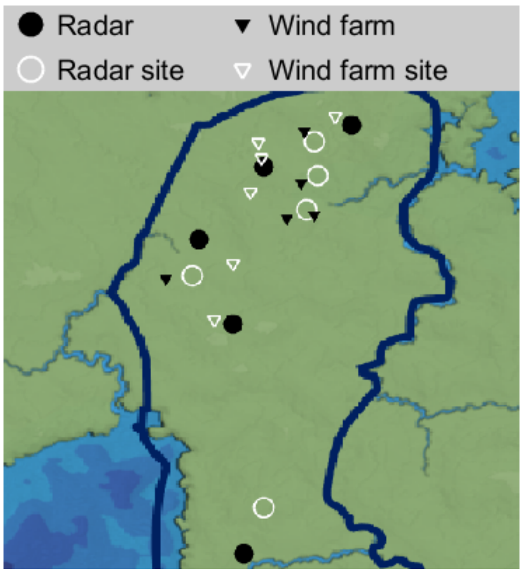
The extended QUB algorithm terminated faster than the QUB algorithm. On average, it halted in the medium problems in 1 hour 33 minutes and 16 computed ASQVs. It used the refinement phase with the local optimization algorithm, since the number of alternatives was larger than 10^5.

The medium problems were also solved using the refinement phase for fewer than 10^5 alternatives. The termination took on average 2 hours 36 minutes with 17 computed ASQVs. The termination was slower than with the local optimization algorithm, since the QUB of every alternative was computed, which took approximately 1 hour.

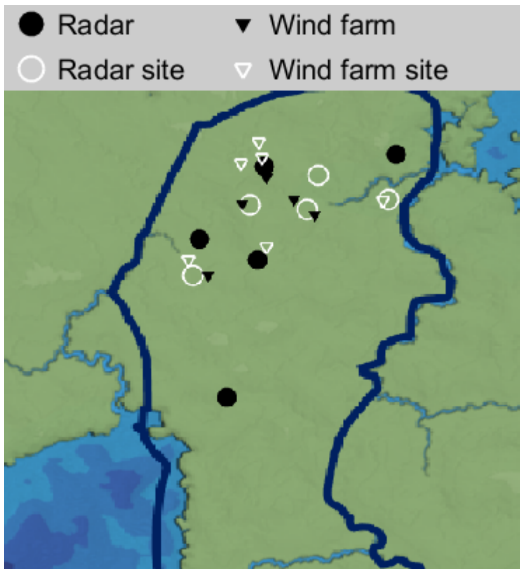
The alternatives obtained with the different refinement phases were similar. Both refinement phases found the same alternative in medium problems 1, 2, and 3. Figure 21 shows these alternatives. However, in medium problems 4 and 5, the returned alternatives differed in the wind farm placements. In problem 4, the ASQVs differed by 2×10^{-3}, in favor of the refinement phase for fewer alternatives. Figure 22 shows the differing alternatives in medium problem 4. Conversely, in problem 5, a 2×10^{-4} higher ASQV was obtained with the local optimization refinement phase. Figure 23 illustrates the differences between alternatives in medium problem 5. The differences in the returned alternatives can be explained by the assumption violations, the randomness of RRHC, and that the refinement phases used were different. The algorithm returning similar alternatives with both refinement phases suggests that, with over 10^5 alternatives, the refinement phase yields near-optimal results despite lacking optimality guarantees. Since both refinement phases returned similar alternatives, but the refinement phase for problems with more than 10^5 alternatives completed one hour faster, the 10^5 threshold was selected for the extended QUB algorithm.



(a) Medium problem number 1

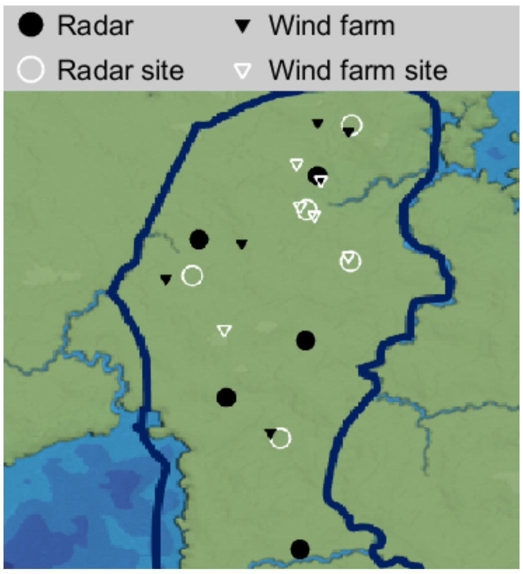


(b) Medium problem number 2.

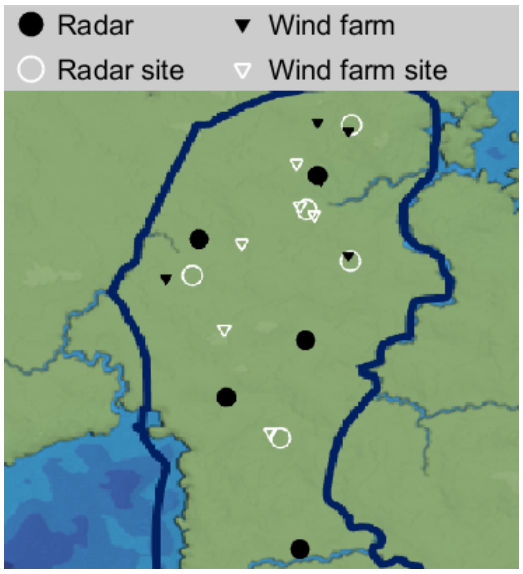


(c) Medium problem number 3.

Figure 21: The alternatives that the algorithms returned in medium problems 1, 2, and 3.

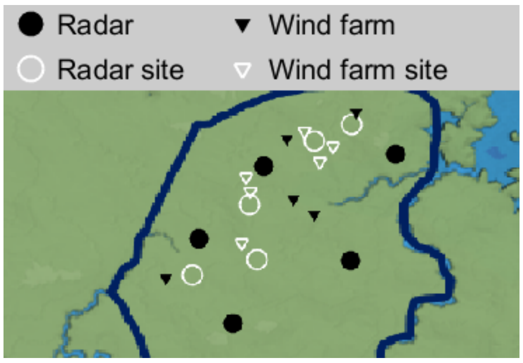


(a) The refinement phase for less than 10^5 alternatives.

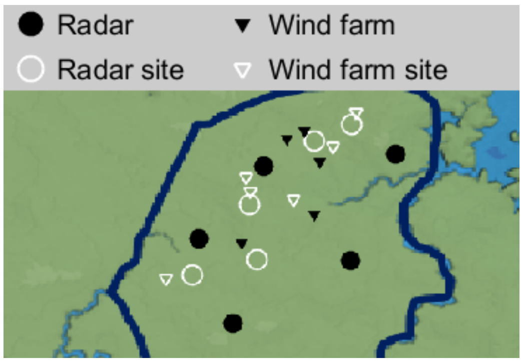


(b) The refinement phase for more than 10^5 alternatives.

Figure 22: The alternatives that the extended QUB algorithm returned with the different refinement phases in medium problem 4.



(a) The refinement phase for less than 10^5 alternatives.



(b) The refinement phase for more than 10^5 alternatives.

Figure 23: The alternatives that the extended QUB algorithm returned with the different refinement phases in medium problem 5.

5.2.3 Large problem

The extended QUB algorithm was applied to the large problem. It used the refinement phase for more than 10^5 alternatives and terminated in 25 hours by computing the ASQV of 90 alternatives. This result shows that the extended QUB algorithm is suitable for larger problems. Figure 24 shows the returned alternative.

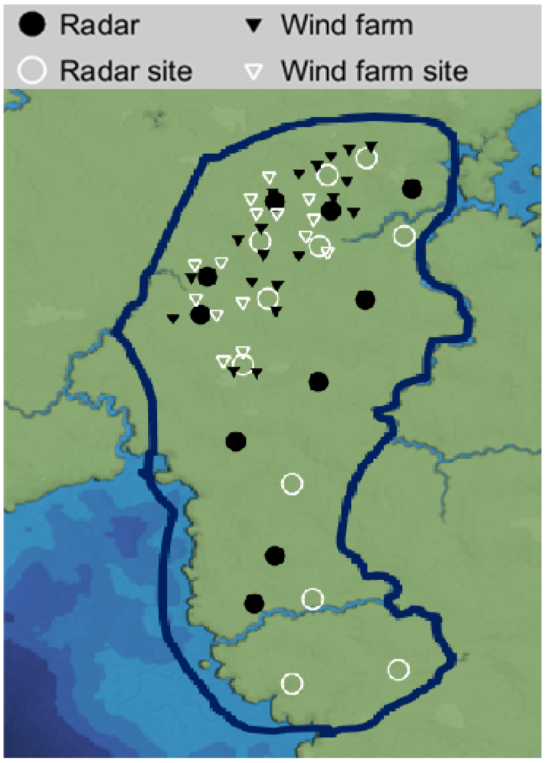


Figure 24: The alternative returned by the extended QUB algorithm in the large problem.

5.2.4 Impact of assumption violations

In the numerical experiments, assumptions of Section 3.1 were violated. Two causes for the violations were identified. First, the radar coverages determined with the internal method of the algorithms were too small, occasionally causing the radars to detect targets outside their coverages. As seen in Section 3.5, too small radar coverages violate Assumption 2. Second, the computation of the track metric is stochastic, which violates the assumption that the computation of the performance metrics is deterministic.

As described in Sections 3.5 and 4.5, assumption violations can cause the QUB and extended QUB algorithms to terminate before identifying the optimal alternative. This early termination occurs because the QUB of the optimal alternative may fall below its ASQV. Knowing how much lower the optimal alternative's QUB is compared to its ASQV allows assessing how close a returned alternative is to the true optimum. For example, suppose the QUB algorithm returns an alternative with an ASQV of 0.9, and the optimal alternative's QUB is known to be 10^{-4} below its ASQV. Then, the optimal alternative's ASQV cannot be greater than 0.9001. Otherwise, the loop at step 7 of Algorithm 2 would not have terminated, because the QUB of the optimal alternative would have been higher than the maximum computed ASQV (= 0.9). This implies that the returned solution is within 10^{-4} of the optimal ASQV. Therefore, the impact of assumption violations on the numerical experiments is evaluated by estimating how much the optimal alternative's QUB is smaller than its ASQV.

Determining exactly how much the optimal alternative's QUB falls below its ASQV requires identifying the optimal alternative, which involves computing the ASQV for all alternatives. Computing the ASQV for all alternatives is time-consuming for the medium and large problems. Instead of computing the ASQV of all alternatives, the difference between the optimal alternative's QUB and its ASQV is estimated.

The amount by which the optimal alternative's QUB falls below its ASQV is estimated based on how much other alternatives' QUBs fall below their ASQVs. This estimate is justified because the causes of the assumption violations reduce the QUBs of all alternatives by a similar amount. For instance, if a radar's coverage is too small, as in Figure 25, the resulting QUB reduction is of similar magnitude across all alternatives that include a radar at that site. Likewise, stochasticity in the track performance metric lowers QUBs by similar amounts across the alternatives.

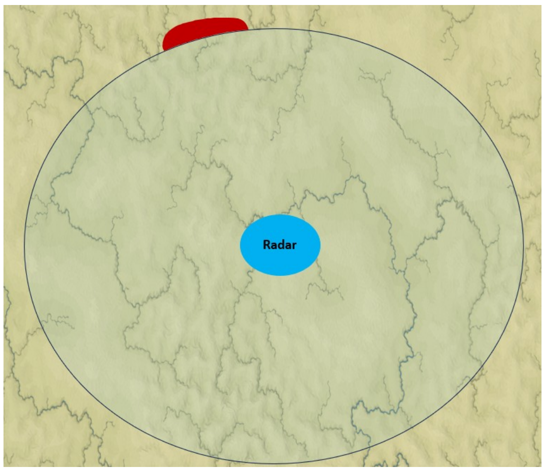


Figure 25: The coverage of a radar depicted in shaded blue, and a region where the radar can detect targets outside its coverage depicted in red.

The amount by which QUBs of alternatives fall below their ASQVs was computed for each algorithm run in the numerical experiments. In each algorithm run, the algorithm computed the ASQVs for a set of alternatives. After each run, the QUB was computed for the same alternatives whose ASQV had been computed during the run. In all runs, the maximum difference by which an alternative's QUB fell below its computed ASQV was no greater than 3×10^{-3}, with the median and mean differences around 10^{-4} in each case. Therefore, it is estimated that the magnitude by which the optimal alternatives' QUBs may have fallen below their ASQVs in the numerical experiments is 3×10^{-3}.

Under the assumptions, the extended QUB algorithm with the refinement phase for less than 10^5 alternatives finds the optimal alternative. Therefore, it is estimated that the extended QUB algorithm provided alternatives in the medium problems within 3×10^{-3} of the true optimum. However, this is an estimate and not a guarantee.

6 Discussions

This thesis developed two algorithms: the QUB algorithm and the extended QUB algorithm. Both algorithms are designed to optimize the placement of radars and wind farms to maximize air surveillance quality. Air surveillance quality refers to how well the radars meet air surveillance objectives and is quantified using the ASQV.

The ASQV is computed via simulation using a computational tool. The computational tool outputs geographically dependent values of performance metrics, which are used for determining the ASQV. Simulating all the values of the performance metrics is computationally intensive. As a result, computing the ASQV for a single alternative takes minutes. When there are many alternatives, identifying the optimal one by evaluating the ASQV of every single one would be prohibitively time-consuming. The QUB algorithms aim to solve this problem.

The QUB algorithms find optimal alternatives faster than an exhaustive search by computing alternative-specific upper bounds for the ASQV. These upper bounds are faster to evaluate than the ASQV. The idea of the algorithms is to iteratively identify alternatives with a high upper bound and determine their ASQV. Once no alternative has an upper bound that exceeds the ASQV of an alternative, the alternative is returned. The QUB algorithm was developed to present this core idea in a simple form. The extended QUB algorithm builds on it by using more efficient search strategies.

The upper bounds are determined using data obtained from computing ASQVs and assumptions based on that radars improve air surveillance, wind farms degrade radar performance, and these effects are geographically constrained. The geographically constrained regions in which radars are assumed to enhance surveillance are referred to as radar coverages. Similarly, the regions in which wind farms are assumed to degrade radar performance are referred to as adverse effect regions. The algorithms internally determine the coverage area for each feasible radar-site pair and the adverse effect region for each radar and wind farm pair. If the assumptions hold, the algorithms return optimal alternatives.

The algorithms were tested through numerical experiments. The goal of these experiments was to demonstrate that the QUB algorithms can identify high-quality alternatives more rapidly than computing the ASQV of every alternative. Next, a summary of the numerical experiments is given. Then, based on the results of the numerical experiments and rest of the thesis, the practical use of the algorithms is discussed. Finally, ideas for future research are offered.

6.1 Summary of the numerical experiments

The numerical experiments demonstrated that the QUB algorithms terminate orders of magnitude faster than computing the ASQV of all alternatives. Additionally, the experiments demonstrated that the algorithms return near-optimal solutions despite violations of the algorithms' assumptions. The algorithms would have returned optimal alternatives if the assumptions were not violated. Moreover, the numerical experiments provided benchmark data on the algorithms' runtimes. The extended QUB algorithm terminated in 25 hours, computing the ASQV of 90 alternatives, when applied to the large problem with 10^{15} alternatives in Section 5.2.3. When the problems were smaller, the extended QUB algorithm required fewer ASQV evaluations. For example, in problems with 10^5 alternatives (Section 5.2.2), it evaluated 16 alternatives on average and solved the problems in 1 hour 33 minutes on average. Small problems with 100 alternatives took on average 44 minutes in Section 5.2.1 with the QUB algorithm, evaluating on average 12 ASQVs.

6.2 On the use of the algorithms

The extended QUB algorithm is recommended for practical use for problems of all sizes due to its faster runtime compared to the QUB algorithm. It reliably finds near-optimal solutions to problems. This claim was justified theoretically in Section 4.4 and empirically through the numerical experiments in Section 5.

The quality of both QUB algorithms' outputs depends on how well the computational tool satisfies the assumptions outlined in Section 3.1. These assumptions include that adding a radar either improves or maintains the values of the performance metrics, while adding a wind farm does not lead to any improvement. The algorithms also assume that the computation of performance metrics is deterministic, meaning that identical inputs produce identical outputs. Furthermore, the algorithms assume that adding or removing a radar only affects performance metric values within its coverage, determined with the internal method of the algorithms. If the internal method identifies the coverage incorrectly, performance metric values can change outside the coverage, violating the assumption. Similarly to the coverages, the algorithms assume that changes to wind farms impact performance metrics only within the adverse effect regions. The adverse effect regions are determined with the internal method of the algorithms, and if the values of the performance metrics change outside these regions, the assumption is violated.

If the computational tool satisfies the underlying assumptions of the algorithms, the extended QUB algorithm returns an optimal alternative when the number of alternatives is below 10^5. This threshold was chosen based on empirical evidence that, beyond 10^5 alternatives, the computational cost of verifying optimality outweighs the potential gains in solution quality. The numerical experiments showed that verifying the optimality resulted in a 2 hours 36 minutes runtime in problems of size 10^5. By instead verifying that the returned alternative cannot be improved by moving a single radar or wind farm, the runtime was 1 hour 33 minutes, with the same quality of the outputs. Therefore, when the number of alternatives exceeds this threshold, the extended algorithm instead returns an alternative whose ASQV cannot be improved by moving a single radar or wind farm, which can be the optimal alternative, but is not guaranteed to be. The QUB algorithm returns the optimal alternative for any number of alternatives under the same assumptions, but requires more time to terminate.

If the computational tool violates the algorithms' assumptions, the returned alternative may not be optimal. Generally, the more closely the tool adheres to these assumptions, the better the output quality.

The spatial overlap between radar coverage areas and adverse effect regions affects the runtime of the QUB algorithms: less overlap results in faster termination. In this thesis, both radar coverages and adverse effect regions are determined as 2D areas by the algorithms. Since radars can detect targets further away at higher altitudes, these regions expand as the maximum altitude of the ASRs increases. Greater region size leads to more overlap, which in turn slows down the algorithm. All numerical experiments were conducted with a maximum altitude of 3000m. Thus, if higher altitudes are considered, the algorithms' runtimes could be longer than reported in the numerical experiments.

6.3 Future research

As discussed in Section 3.5, the assumptions outlined in Section 3.1 were violated in the numerical experiments by insufficient radar coverages, meaning that adding and removing radars changed values of performance metrics outside their coverages determined with the algorithms' internal method. Additionally, the track metric was stochastic. Future work could address these issues by modifying the internal method to produce sufficient radar coverages. This could be done by using the same line of sight method used in this thesis, but enlarging the regions by adding a little buffer. Another option would be to compute radar-specific probabilities of detections with respect to the targets of the ASRs and determine the coverages from the locations where the probability of detection is greater than zero. Moreover, the algorithms' performance could be improved by minimizing the variance in track metric computations.

When computing the ASQV for a single alternative takes several minutes, optimizing over many alternatives becomes inevitably time-consuming. Future work could reduce computation time by using radar performance models with lower computational cost. In particular, combining the simplified models with standard integer programming techniques (see, e.g., Wolsey, 2020) would allow fast optimization with a large number of alternatives. For example, this approach could be used to pre-screen promising alternatives before applying more accurate but slower models. By doing so, the total computational burden of optimization is reduced.

The algorithms could be enhanced to terminate faster in higher altitude scenarios, where radar coverages and adverse effect regions expand and overlap more compared to lower altitude cases. As these regions grow, more ASQV computations are required, resulting in longer runtimes. One approach to reducing the increased computational cost is to modify the algorithms' internal method to produce radar coverages and adverse effect regions in 3D rather than 2D, causing less overlap. As part of this approach, the algorithm could use a 3D location-specific value as a base for the upper bound computations, instead of 2D fulfillment values of ASRs.

This thesis considered the placement of ground-based and monostatic radars. Future work could explore optimizing the placement of other radar types, such as passive radars (see Kuschel et al., 2019). Assumptions similar to those in Section 3.1 could be applied to different radar types, enabling the use of algorithms comparable to those developed in this thesis.

The developed algorithms jointly optimize the placement of radars and wind farms. These algorithms are also suitable for optimizing the placement of radars when the locations of the wind farms are fixed. Additionally, they can be used to optimize the placement of wind farms when the radar locations are fixed.

7 Conclusions

Wind farms are being constructed in increasing numbers. However, they have adverse effects on air surveillance radars. This thesis studied the optimization of radar and wind farm placement to minimize these effects.

The thesis's goal was to present methods for optimally placing radars and wind farms to maximize air surveillance quality, quantified using accurate but time-consuming simulations. The thesis focused on placement problems where the number of alternatives was 10^2 to 10^{15}. Finding the optimal alternative by simulating every single one would take hours to 10^{10} years, depending on the number of alternatives in the problem.

To speed up the search for the optimal alternative, this thesis introduced two domain-specific algorithms: the QUB and extended QUB algorithms. The QUB algorithm is a simple method intended to illustrate the main algorithmic idea, which is to find the optimal alternative by computing upper bounds for the air surveillance quality of alternatives. These upper bounds are computed using assumptions based on the impact of radars and wind farms on air surveillance and previous simulation results. The alternative with the maximum upper bound is iteratively identified and evaluated with the expensive simulations. Once no upper bound exceeds the air surveillance quality of an alternative, the optimal alternative is found and returned. The extended algorithm builds on the same idea but uses more efficient search strategies for identifying alternatives with high upper bounds.

The algorithms were analyzed theoretically and tested through numerical experiments. The better the simulations adhere to the algorithms' assumptions, the better alternatives the algorithms output. If the simulations completely adhere to the assumptions, the extended QUB algorithm returns optimal solutions in problems with fewer alternatives than 10^5. In problems with more than 10^5 alternatives, it outputs an alternative that cannot be improved by moving a single radar or wind farm, which can be the optimal alternative, but is not guaranteed to be. The QUB algorithm always returns the optimal alternative, but the QUB algorithm is slower than the extended QUB algorithm, regardless of the number of alternatives. The experiments demonstrated that the algorithms terminate within hours, when the simulation of every alternative would take years. The algorithms returned near-optimal solutions despite deviations from their assumptions. The extended QUB algorithm is recommended for practical use as it is faster and returns equally good radar and wind farm placements as the QUB algorithm.

This thesis fills a gap in the existing literature by presenting radar and wind farm placement optimization algorithms for situations when there are many alternative ways to place the radars and wind farms. The optimization relied on accurate but time-consuming simulations, ensuring the reliability of the solutions. Furthermore, the thesis serves as a case study of developing problem-specific surrogate-assisted optimization algorithms. It presents a novel algorithmic idea for optimization with computationally intensive simulation. The developed algorithms facilitate the coexistence of air surveillance and wind farms while maximizing the effectiveness of ground-based monostatic air surveillance radars.

References

Ahn, B. S. (2011). Compatible weighting method with rank order centroid: Maximum entropy ordered weighted averaging approach. *European journal of operational research*, *212*(3), 552–559.

Audet, C., & Hare, W. (2017). *Derivative-free and blackbox optimization*. Springer.

Baptista, R., & Poloczek, M. (2018). Bayesian optimization of combinatorial structures. *Proceedings of the 35th International Conference on Machine Learning*, 462–471.

Bartz-Beielstein, T., & Zaefferer, M. (2017). Model-based methods for continuous and discrete global optimization. *Applied Soft Computing*, *55*, 154–167.

Chugh, T., Sindhya, K., Hakanen, J., & Miettinen, K. (2019). A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Soft Computing*, *23*, 3137–3166.

Cranmer, A., Baker, E., Liesiö, J., & Salo, A. (2018). A portfolio model for siting offshore wind farms with economic and environmental objectives. *European Journal of Operational Research*, *267*(1), 304–314.

De la Vega, D., Matthews, J. C., Norin, L., & Angulo, I. (2013). Mitigation techniques to reduce the impact of wind turbines on radar services. *Energies*, *6*(6), 2859–2873.

Dhillon, S. S., & Chakrabarty, K. (2003). Sensor placement for effective coverage and surveillance in distributed sensor networks. *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, *3*, 1609–1614.

Ek, K., & Persson, L. (2014). Wind farms—where and how to place them? a choice experiment approach to measure consumer preferences for characteristics of wind farm establishments in sweden. *Ecological economics*, *105*, 193–203.

Fu, M. C. (Ed.). (2015). *Handbook of simulation optimization*. Springer.

Harju, M., Liesiö, J., & Virtanen, K. (2019). Spatial multi-attribute decision analysis: Axiomatic foundations and incomplete preference information. *European Journal of Operational Research*, *275*(1), 167–181.

Huttunen, R., Sallinen, A., Hämäläinen, K., Passoja, K., & Palander, S. (2024). *Itäisen suomen tuulivoimarakentamisen työryhmän loppuraportti* [Final report of the working group on wind power construction in Eastern Finland]. Työ- ja elinkeinoministeriön julkaisuja 2024:28. Retrieved from https://urn.fi/URN:ISBN:978-952-327-566-9.

Jin, Y. (2011). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, *1*(2), 61–70.

Joensuu, K., Väyrynen, L., Tolppanen, J., Karhu, L., Salmi, T., Hartikka, S., Leino, L., Viljanen, J., Smids, S., Hujanen, A., Sipilä, M., & Huuskonen, A. (2021). *Tuulivoimarakentamisen edistäminen: Keinoja sujuvaan hankekehitykseen ja eri tavoitteiden yhteensovitukseen* [Advancing wind power construction:

Means for streamlining of project development and for coordination of various objectives]. Valtioneuvoston selvitys- ja tutkimustoiminnan julkaisusarja 2021:51. Retrieved from http://urn.fi/URN:ISBN:978-952-383-354-8.

Kuschel, H., Cristallini, D., & Olsen, K. E. (2019). Tutorial: Passive radar tutorial. *IEEE Aerospace and Electronic Systems Magazine*, *34*(2), 2–19.

Lahti, P. (2022). *Compensation of adverse effects of wind farms on air surveillance capability using spatial decision analysis* (Master's thesis). Retrieved from https://urn.fi/URN:NBN:fi:aalto-202206194062.

Larson, J., Menickelly, M., & Wild, S. M. (2019). Derivative-free optimization methods. *Acta Numerica*, *28*, 287–404.

Malczewski, J., & Rinner, C. (2015). *Multicriteria decision analysis in geographic information science*. Springer.

Oh, C., Tomczak, J., Gavves, E., & Welling, M. (2019). Combinatorial bayesian optimization using the graph cartesian product. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, *32*, 2910–2920.

Papalexopoulos, T. P., Tjandraatmadja, C., Anderson, R., Vielma, J. P., & Belanger, D. (2022). Constrained discrete black-box optimization using mixed-integer programming. *Proceedings of the 39th International Conference on Machine Learning*, 17295–17322.

Red Blob Games. (2018). Mapgen4 [Accessed: 2.4.2025]. Retrieved from https://www.redblobgames.com/maps/mapgen4/.

Saidur, R., Rahim, N. A., Islam, M. R., & Solangi, K. H. (2011). Environmental impact of wind energy. *Renewable and sustainable energy reviews*, *15*(5), 2423–2430.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & De Freitas, N. (2015). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175.

Simon, D. (2013). *Evolutionary optimization algorithms*. John Wiley & Sons.

Sunak, Y., Höfer, T., Siddique, H., Madlener, R., & De Doncker, R. W. (2015). *A gis-based decision support system for the optimal siting of wind farm projects*. Universitätsbibliothek der RWTH Aachen Aachen, Germany.

Tema, E. Y., Sahmoud, S., & Kiraz, B. (2024). Radar placement optimization based on adaptive multi-objective meta-heuristics. *Expert Systems with Applications*, *239*, 122568.

Virtanen, M. (2024). *Evaluation of quality of air surveillance using spatial multi-criteria decision analysis* (Master's thesis). Retrieved from https://urn.fi/URN:NBN:fi:aalto-202411217306.

Wolsey, L. A. (2020). *Integer programming*. John Wiley & Sons.

Yang, Y., Yi, W., Zhang, T., Cui, G., Kong, L., Yang, X., & Yang, J. (2015). Fast optimal antenna placement for distributed mimo radar with surveillance performance. *IEEE Signal Processing Letters*, *22*(11), 1955–1959.