Enhanced Optimal Beacon Placement for Indoor Positioning: A Set Variable Based Constraint Programming Approach

Sven Löffler, Ilja Becker, Carlo Bückert and Petra Hofstedt
Programming Languages and Compiler Construction, Brandenburg University of Technology Cottbus - Senftenberg, Konrad-Wachsmann-Allee 5, 03046 Cottbus, Germany
{*sven.loeffler, ilja.becker, carlo.bueckert, hofstedt*}*@b-tu.de*

Keywords: Beacon Placement, Constraint Programming, Indoor Positioning, Trilateration, Optimization Algorithms, Decision Support Systems.

Abstract: Indoor localization is of increasing importance in various environments, including hospitals, retirement homes, and emergency situations. To achieve efficient and accurate positioning of mobile individuals indoors, the optimized distribution of sensors is crucial. The task of manually placing beacons (sensors) for indoor positioning in a building can be challenging and time-consuming. Several researchers have tackled this issue using different algorithms and considering various use cases. In our previous work (Löffler et al., 2022) at the ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2022), we introduced a novel approach that leverages constraint programming with exclusively Boolean variables to efficiently place Bluetooth Low Energy (BLE) beacons in indoor scenarios. We evaluated the quality of our results by comparing them against manually optimized beacon placement and assessing their performance in a real-world building. This paper extends the findings of (Löffler et al., 2022) by introducing a new constraint-based approach that incorporates only set variables and Boolean variables, a more elaborate and balanced evaluation, i.e. on further buildings, and certain refinements of our overall method.

1 INTRODUCTION

Indoor localization is an increasingly important topic, particularly in hospitals, retirement homes, and highly industrialized work areas. The ability to quickly locate patients, disoriented individuals, or injured persons can be crucial in emergency situations, potentially saving lives. For example, in many medical facilities individuals are currently brought to the treatment area well in advance of their scheduled appointments to ensure seamless execution of treatments without time gaps. However, from the patient's perspective, the resulting waiting time is often extremely frustrating and can even have negative health effects. In nursing homes and retirement facilities, where individuals have freedom of movement, staff members often face challenges in locating patients in a timely manner for treatment. By employing indoor tracking systems, individuals can be located and promptly directed to their destinations, reducing treatment waiting times and improving overall circumstances. Another application of indoor location services is assisting customers in navigating large public or private buildings, such as libraries, parking lots, or office complexes as well as workers in large industrial plants, e.g. in rescue situations.

The availability of smaller Bluetooth Low Energy (BLE) beacons with extended battery lifetimes, along with the integration of Bluetooth and GPS technologies in even the most affordable smartphones, enables rapid and reliable determination of positions in various scenarios. Given that GPS signals are typically unavailable or unreliable inside buildings, indoor positioning plays a crucial role in facilitating seamless transition and navigation within such structures. Indoor positioning applications encompass indoor navigation, asset tracking, people or personnel tracking, and more. Among the prevalent techniques used in indoor positioning algorithms, BLE beacons and signal strength measurements are commonly employed for triangulation, trilateration, or simple proximity-based approaches. This paper focuses on trilateration, a measurement method for determining the position of a point based on its distances to three reference points. The placement of beacons for trilateration should strive for optimality, aiming for minimal beacon usage while maintaining efficacy. The placement and overall quantity of the beacons should be as close to optimal as possible for these methods. Moreover, beacon placement should be cost-efficient and unobtrusive.

Presently, beacon placement is often performed manually. Manual beacon placement is not only time-consuming but also prone to errors. Even after extensive testing, there is no guarantee of complete coverage throughout the entire building, nor certain knowledge of a minimum (or acceptably small) number of beacons. With our new constraint-based approaches, assuming appropriately pessimistic parameter choices, we can ensure complete coverage of all positions with a minimum number of beacons at the same time. Our model can also be used to verify manually created beacon placements for their coverage adequacy.

In this paper, we present a new method to streamline and enhance the beacon placement process, leveraging constraint optimization and programming techniques to compute optimal beacon positions suitable for common indoor positioning algorithms. We have modeled a constraint problem based on set variables and boolean variables, which employs highly effective channeling constraints.

The remainder of this article is organized as follows. Section 2 provides an overview of constraint programming fundamentals essential for our proposed method. Section 3 reviews prior research on BLE beacon placement and indoor localization. Section 4 describes the beacons, hardware, and software used in developing our method. The subsequent section (Section 5) outlines the developed constraint problem for beacon placement. Section 6 presents experimental results obtained from real-world scenario evaluations. Finally, we conclude the paper and discuss potential future improvements to our new method (Section 7).

2 BASICS OF CONSTRAINT PROGRAMMING

Constraint programming (CP) is a powerful method for declaratively modeling and solving NP problems. Common research areas in CP include rostering, graph coloring, optimization, and satisfiability (SAT) problems (Marriott, 1998). In this section, we provide the necessary definitions of constraint programming for our approach, which can be summarized in two steps:

1. Declarative representation of a problem as a constraint model.
2. Independent solving of the constraint model by a solver.

The CP user's responsibility is to model the application problem with constraints (1) and initiate the solver, which runs independently like a black box (2).

We begin by introducing constraints based on finite domain variables before expanding this concept to include set variables.

Finite Domain Variables and Constraints

A *constraint* (X,R) consists of a relation R and an ordered set of variables X on which the relation is defined (Dechter, 2003a). Examples include $(\{x,y\},x > y)$, $(\{x,y,z\},x*y=z)$, or $(\{A,B\},A \rightarrow B)$. In this paper, we only specify the relation of a constraint since the variables used in a constraint are clearly identifiable.

A *constraint satisfaction problem (CSP)* is a 3-tuple $P=(X,D,C)$, where $X=\{x_1,x_2,\ldots,x_n\}$ is a set of variables, $D=\{D_1,D_2,\ldots, D_n\}$ is a set of finite domains where D_i is the domain of x_i, and $C=\{c_1,c_2,\ldots,c_m\}$ is a set of constraints involving one or more variables from X (Apt, 2003a).

A *solution* to a CSP is an assignment of values d_j from their corresponding domains D_i to the variables x_i, satisfying all constraints. A *constraint optimization problem (COP)* is an extension of a CSP, where an optimization variable x_{opt} identified as such will be minimized or maximized.

The following COP 1 is an example for a COP, which describes the problem of finding a rectangle $a \times b$ with only integer values $\{1,2,3,4,5\}$ for a and b, which has a maximum floor area while at the same time the perimeter must not be greater than 15.

COP 1 $=(X,D,C)$ with $X=\{a,b,A,P\}$, $D=\{D_a=D_b=\{1,2,3,4,5\},D_A=\{1,2,...,25\},D_P=\{1,2,...,15\}\}$ and the constraints $C=\{(a*b=A),(2*a+2*b \leq P)\}$. The optimization variable is A and should be maximized. An optimal solution of this COP is $a=4$, $b=3$, $A=12$, and $P=14$.

A specific constraint relevant to the upcoming work is the *count* constraint. The constraint $count(X,occ,v)$ limits the variables of the set X such that the value v only occurs occ times (GCC, 2022; van Hoeve and Katriel, 2006). For example, the constraint $count(\{x_1,x_2,x_3,x_4\},3,1)$ guarantees, that the value 1 occurs exactly 3 times in the variables x_1,x_2, x_3 and x_4. Instead of a constant, we also allow a set of constants as input for the occurrence in the *count* constraint. In this case the occurrence must be equal to a value in the set. For example $count(\{x_1,x_2,x_3,x_4,x_5\},\{2,3\},1)$ is only satisfied, if two or three of the variables are assigned the value 1.

To solve constraint problems (CSPs and COPs), so-called solvers usually use backtracking search nested with propagation. Popular finite domain solvers are among others Google OR tools (Perron and Furnon, 2023), Gecode (Christian Schulte, 2019), JaCoP (Kuchcinski and Szymanek, 2013) or the Choco solver (Prud'homme et al., 2017), with the latter beeing used here. More information about solvers and their mode of operation can be found in (Rossi et al., 2006; Dechter, 2003b; Apt, 2003b).

Set Variables and Constraints

So far, we have only considered integer variables that can take on a single value. However, going forward, we will also allow set variables that can be associated with a collection of values. A set variable x is a variable that has a discrete domain $D(x) = [lb(x), ub(x)]$, where $lb(x)$ is a set of values representing the lower bound and $ub(x)$ is a set of values representing an upper bound for x. Thus, the domain of a set variable consists mandatory elements (exactly $lb(x)$) and possible elements beyond that, i.e. the elements of $ub(x) \setminus lb(x)$. The value assigned to x should be a set $s(x)$ such that $lb(x) \subseteq s(x) \subseteq ub(x)$ (van Hoeve and Katriel, 2006). An example of a set variable is x^S with a domain $D(x^S) = \{\{\}, \{0,1,2\}\}$. This indicates that the set variable can take the following valid assignments: $\{\}$, $\{0\}$, $\{1\}$, $\{2\}$, $\{0,1\}$, $\{0,2\}$, $\{1,2\}$, $\{0,1,2\}$. Thus, the variable x^S is not required to contain any value, but it can include any number of values from the set $\{0,1,2\}$. If the lower bound were set to $\{1\}$, it would mean that all assignments must include the value 1: $\{1\}$, $\{0,1\}$, $\{1,2\}$, $\{0,1,2\}$.

Set variables are somtimes already considered in finite domain constraint solvers like Choco. It is also possible to handle set variables as set of Boolean variables. In this case for each value d_j of each variable x_i a Boolean variable $x^B_{i,j}$ is created, which represent whether the original set variable x_i contains value d_j ($x^B_{i,j} = True$) or not ($x^B_{i,j} = False$).

Regardless of how set variables are handled by the solver, they enable us to utilize additional useful methods and constraints. For example, the method $setCard(IntVar\ c)$ sets the cardinality of a set var equal to an integer variable c. On the other hand, the $boolsChanneling(B = [x^B_1, x^B_2, x^B_3, ..., x^B_n], x^S)$ constraint binds the set variable x^S to the Boolean variables $x^B_1, x^B_2, x^B_3, ..., x^B_n$, as described before ($i \in x^S \Leftrightarrow B[i] = x^B_i = True$).

3 RELATED WORK

In this section, we delve into current research in the areas of sensor technology, automatic beacon placement, and indoor positioning.

3.1 Different Sensor Technology

There are two primary approaches to indoor localization: infrastructure-based and infrastructure-less methods (Taskan and Alemdar, 2021). Infrastructure-less methods rely on environmental features (sound, light, magnetic fields, or smartphone sensors) to obtain location fingerprints. Infrastructure-based methods leverage pre-installed visual sensors or various wireless technologies like ZigBee (Fang et al., 2012), WiFi, Ultra-Wideband (UWB), RFID (Zhang et al., 2016), and BLE (Subedi and Pyun, 2020). Infrastructure-based indoor positioning systems can be costly due to required methodologies or expensive hardware.

Among infrastructure-based technologies, BLE has gained widespread adoption in ubiquitous computing and IoT applications due to its low power consumption and cost-effectiveness (Kuxdorf-Alkirata et al., 2019). BLE beacons are portable, easy to deploy, and offer high positioning accuracy. While ZigBee consumes less energy, its support on smartphones is limited. WiFi has broad support but higher energy requirements compared to BLE (Subedi and Pyun, 2020). UWB, an emerging technology, is not yet supported on smartphones.

3.2 Automatic Beacon Placement

A significant amount of research has been conducted on the placement of beacons for indoor positioning algorithms. In the following, we outline the most notable approaches.

The first step in beacon placement is determining a metric to assess the quality of the placement. One commonly used metric is the *Geometric Dilution of Precision (GDOP)*. Originally used for navigation satellites, GDOP quantifies the spread of measured values and depends on the relative positions of the satellites and the observer. In the work by (Rajagopal et al., 2016), they introduce a method to adapt GDOP for indoor spaces. This augmented GDOP is then utilized in a toolchain to evaluate various beacon placement algorithms. The study reveals a significant reduction in the number of required beacons compared to standard trilateration techniques.

One category of algorithms for position calculation relies on the *Time-of-Flight (ToF)* of signals transmitted between the beacons and the target device. In the research conducted by (Wang et al., 2019), beacon position optimization for such algorithms is explored. They propose a greedy algorithm that initially places $O(OPT \ln(m))$ beacons, where m represents the number of discrete location points in the region and OPT denotes the size of the optimal solution. Furthermore, a random sampling algorithm is introduced, which reduces the number of required beacons to $O(OPT \ln(OPT))$. These algorithms, on average, place between 6% to 23% fewer beacons compared to prior works.

The study conducted by (McGuire et al., 2021) focuses on the self-localization of autonomous vehicles in scenarios where traditional methods such as GPS positioning are not available. In contrast to the previously mentioned approaches, they utilize the *Angle-of-Arrival* (AoA) for position calculation. The inclusion of the heading angle sets this method apart from others and adds complexity to the resulting optimization problem. The paper first presents the determinant of the Fisher information matrix for an arbitrary number of beacons. Subsequently, the optimal angular separation for three beacons is analytically derived. The optimality of the solution is then demonstrated through numerical simulations.

In a different approach, (Sharma and Badarla, 2018) adopts a unique strategy by considering the placement domain of beacons as a grid of candidate locations on the surfaces of ceilings and walls in the target indoor environment. They formulate the error propagation resulting from the geometric arrangement between anchor beacons and target devices as an optimization objective. To minimize the total beacon count while satisfying the GDOP constraint mentioned earlier, they employ *Mixed Integer Linear Programming (MILP)*. The proposed method was compared against the conventional linear placement of beacons, which assumes a planar geometry between device and beacon locations. The results demonstrated an improvement in the minimum GDOP while maintaining the same number of placed beacons.

However, to the best of our knowledge, no other work has employed a comparable specific constraint programming approach for beacon placement using solely Boolean variables or set variables.

3.3 Indoor Positioning

One of the main use cases for an automatic placement of beacons in a building is indoor positioning. Hence this section examines some of the current work regarding this use case.

Many of these systems also utilize information obtained from nearby BLE beacons for indoor positioning. One such method is developed by (Tomažič and Škrjanc, 2021). They introduce a novel visual-inertial localization algorithm that automates the collection of Bluetooth data required for positioning. This data is subsequently utilized in a constrained nonlinear optimization algorithm. The developed algorithm is implemented on a smartphone, enabling real-time determination of beacon locations and the creation of path loss models. The paper highlights the advancement of their innovation by allowing users to assess if sufficient data has been computed for achieving the desired positioning accuracy.

A distinct approach was explored by (Grottke and Blankenbach, 2021). Their research employs an evolutionary optimization strategy to implement an indoor positioning algorithm, utilizing the Wireless Local Area Network (WLAN), Bluetooth (BLE), and the smartphone's Inertial Measurement Unit (IMU). Each new source of information modifies the probability values of particles through a probability-map-based approach. Their results showed a reduction of the maximum position error 31%, while the RMSE and the 95-percentile positioning errors could be reduced by 34% and 52% respectively.

The authors of (Amsters et al., 2019) conducted a comprehensive evaluation of efficient indoor positioning systems, considering factors such as cost and accuracy. Their findings indicated significant advancements in the field of indoor positioning. Simple indoor positioning systems, based on cameras, can be obtained for a few hundred dollars, while higher accuracies can be achieved with technologies like *HTC Vive* at a slightly higher cost. However, these systems may not be scalable for larger environments. To address this, the authors recommended systems such as *Marvelmind* or *Pozyx*, which utilize ultrasound ranging and Ultra Wide Band beacons, respectively. Notably, none of the recommended efficient systems relied on BLE beacons, underscoring the need for further improvement in positioning using this technology.

4 EXPERIMENTAL SETUP

In this section we briefly examine the hardware that was used to test the beacon placement in buildings and introduce the software we used to implement our algorithm. Finally, in this section we explain how our input building plans look like. In order to be comparable with (Löffler et al., 2022), we made no changes to the setting, but simply included additional floor plans in the evaluation.

4.1 Hardware and Software

The system uses BLE beacons from the company *blukii* called *Smart Beacon Go Mini*. The beacons are based on Bluetooth 4.2 (Bluetooth Low Energy), use a *Texas Instruments CC2640* controller and are powered by a *CR2032* battery (providing up to a half year of runtime when using a 1 second advertising interval). The beacons support the iBeacon and Eddystone protocols and have a configurable advertising interval of 0.1 to 1 second while providing a range of up to 50 meters.

Different smartphones were used to check the signal strengths in the building from (Löffler et al., 2022) at different locations and to make test and calibration measurements. The main smartphone used for taking measurements was a *Google Pixel 3a* running Android 12 with Bluetooth 5.0 capabilities.

The software for calculating the optimal beacon placement was written in Java and with the help of the Choco solver (Prud'homme et al., 2017) library. No further software was necessary. It is possible to replace Java and the Choco solver by any other programming language and constraint solver. The method presented in the following Section 5 is independent of the used software.

4.2 Building Plans

The building plans for the floors of the considered building were available in the PDF file format. As an extension to (Löffler et al., 2022), we considered 3 more floor plans (see Figures 1 to 3) from typical school buildings in East Germany as tests for the subsequent beacon positioning. We converted the building from (Löffler et al., 2022) and the three new buildings to PNG files with 1,000,000 to 16,000,000 pixels, such that every pixel represents an area of 4cm by 4cm. We differentiate between different wall types and areas within the floor plans: massive walls, drywalls, triple glazed windows, free spaces which must be covered by beacons and free spaces which do not need beacon coverage. The different wall types have an influence on the range of the individual beacons and are considered when later calculating the positions of the beacons.

The selected floor plans have deliberately been chosen to be highly diverse. Thus, the floor plan of the school in Dresden (Figure 2) consists of only one building. In contrast, the schools in Dahlewitz (Figure 1) and Potsdam (Figure 3) consist of two buildings, which are connected by a thick corridor (Dahlewitz) or three small corridors (Potsdam). We have chosen the same resolution of 4 cm by 4 cm for the three new

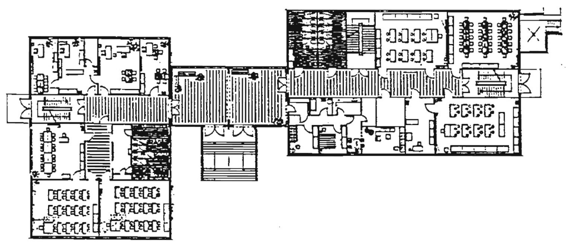
Figure 1: The floor plan of a school in Dahlewitz (Sch, 1999).

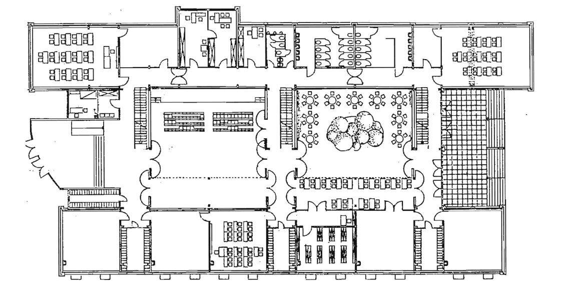
Figure 2: The floor plan of a school in Dresden (Sch, 1999).

floor plans as the floor plan of the building in (Löffler et al., 2022). Similarly, all other parameters are selected in a consistent manner.

5 METHOD

In this section, we briefly explain the approach from (Löffler et al., 2022) and elaborate on our new constraint modeling.

5.1 The General Approach

Figure 4 gives an overview over the developed process which is explained in the following subsections.

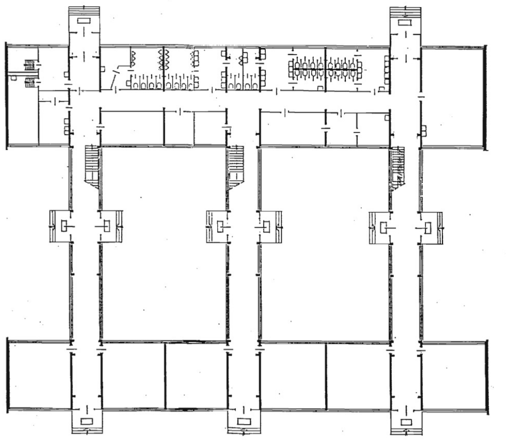
Figure 3: The floor plan of a school in Potsdam (Sch, 1999).

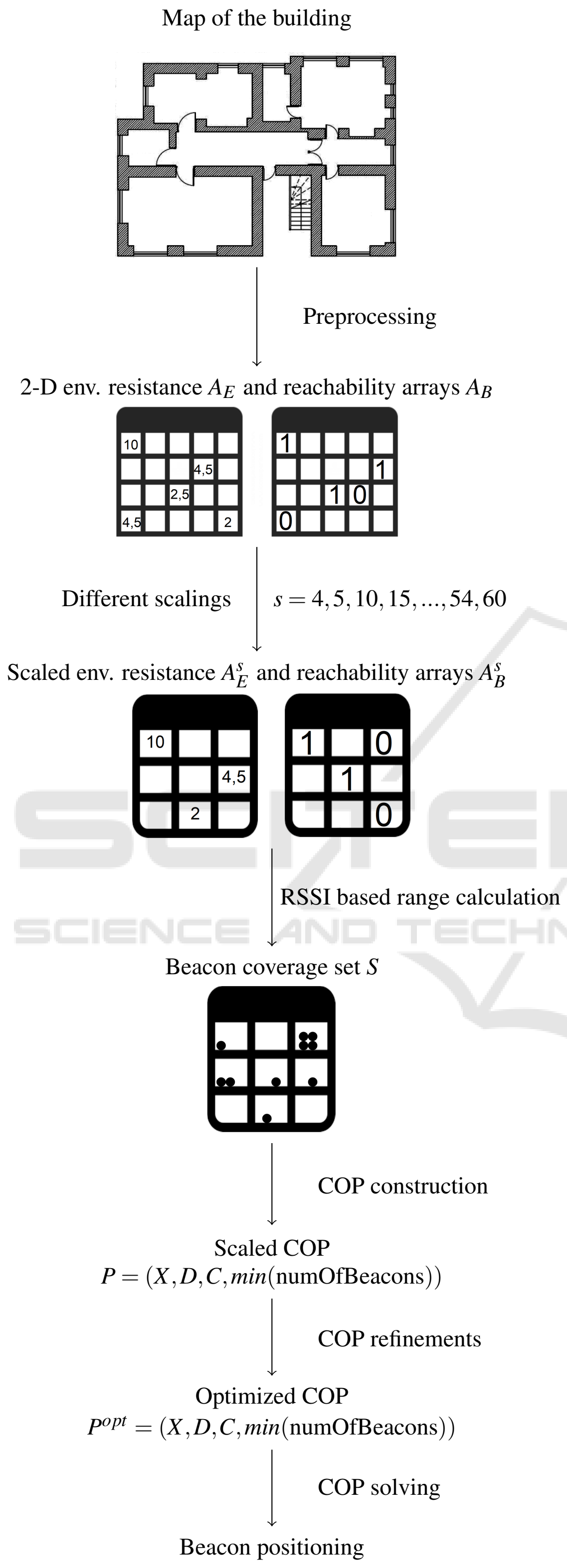
Map of the building
Preprocessing
2-D env. resistance A_E and reachability arrays A_B
10
4,5
2,5
4,5
2
1
1
1 0
0
Different scalings
$s = 4, 5, 10, 15, ..., 54, 60$
Scaled env. resistance A_E^s and reachability arrays A_B^s
10
4,5
2
1 0
1
0
RSSI based range calculation
Beacon coverage set S
COP construction
Scaled COP
$P = (X, D, C, min(\text{numOfBeacons}))$
COP refinements
Optimized COP
$P^{opt} = (X, D, C, min(\text{numOfBeacons}))$
COP solving
Beacon positioning

Figure 4: Overview of the beacon positioning process from (Löffler et al., 2022).

First, the input building plan is preprocessed to capture all relevant wall properties, including wall thicknesses and wall types. This process generates two arrays, one (A_B) reflecting whether a region should be covered by beacons (areas within the building) or not (areas outside the building, walls, etc.), and a second one (A_E) indicating the environmental resistance for beacon signals in each region (environment factor E). These arrays are then scaled to different sizes. Different scalings ensure whether the input image is accurately reconstructed or abstracted. Without scaling, each 4cm x 4cm pixel would be directly taken into account. By scaling, multiple pixels are merged together. Depending on the scaling, the resulting COP arises with a varying number of variables and constraints, which needs different solution times. The coverage of each scaled region by placing a beacon in each area is determined (Beacon coverage set S) by use of Equation 1 and serves as the basis for the Constraint Optimization Problem (COP). Before solving the COP, various refinements are performed on it to minimize the solution time. Finally, the COP is solved with the objective of minimizing the total number of beacons used.

$$distance = 10^{M_1 - \frac{RSSI}{10 * E}} \tag{1}$$

The COP modelling starts with a very large scaling factor, resulting in a low level of detail and a small generated COP. This COP is solved considering a specific time limit, and the best solution found is recorded. Subsequently, the scaling factor is reduced, and a new COP is created. The new COP, due to its higher resolution, exhibits a higher level of detail but also generates a larger COP. Since all calculations resulting from scaling are rounded pessimistically, COPs with a low grid density are the quickest to solve due to its small size. However, it may lead to inferior solutions compared to COPs with higher grid density, which, due to a higher number of variables and constraints, requires more time for solution computation. This approach allows for quickly finding a good solution and, over time, discovering even better solutions.

In comparison to (Löffler et al., 2022), it turned out that we were overly pessimistic by introducing a dynamic component in the calculation of the environmental factor E for the RSSI value. Thus, we only consider static values for E: 2 (freeSpaces), 2.5 (drywalls), 4.5 (massiveWalls), and 10 (glass) and removed the dynamic part which depends on the wall thickness.

The determination of the environmental factor remains pessimistic, as we always consider the highest occurring value along a given path. For instance, if there is a massive wall ($E = 4.5$) between the current position and a beacon, while the rest of the space is unobstructed ($E = 2$), the value of 4.5 will still be assumed for the entire distance in the environmental factor calculation. The constant pessimistic estimation ensures that every calculated solution corresponds to a reliable real-world solution. For more details about the general approach, please refer to (Löffler et al., 2022).

5.2 The Constraint Optimization Problem

We use constraint programming to mathematically model and optimize the beacon placement. Therefore, we create a COP that describes the conditions for the beacon placement (using trilateration), i.e. solutions of the COP are all possible placements of beacons. An optimization is then carried out over this very large number of variants. This is realized by adding an objective function ($minimize(x_{count})$), such that we get a COP.

In (Löffler et al., 2022) a COP based on boolean variables and count constraints was created. This paper presents a new approach which is based on set variables, boolean variables, and channelling constraints between them. The aim of the COP is to achieve a minimum number of beacons while completely covering every point in the building with (at least three) radio signals. The new COP is described below and given in Figure 5.

For our set variable based COP we work with an additional grid $G3$ which represents the input map. Each grid element in $G3$ has a size of 3m by 3m. Therefore, the beacons we want to position can only be placed at locations that are at least three meters apart horizontally and vertically from each other. Keep in mind that the original map can have a pixel size of 4 cm by 4 cm. This means that a beacon can only be placed at every 75th position in both horizontal and vertical directions ($0.04m * 75 = 3m$). So, while the size of our scaled map is determined by the scaling factor s and has $n \times m$ grid elements, the grid $G3$ always has a size of $n^{G3} \times m^{G3}$, where n^{G3} is the length of the building (in meter) divided by 3 and m^{G3} is the width of the building (in meter) divided by 3. The feasible positions for beacon placement, denoted as $S_{x,y}$, only need to contain positions that fall within our grid $G3$.

For each position in the grid $G3$, we create a boolean variable $x_{i,j}$ with a domain $D_{i,j} = \{0,1\}$. Additionally, we have an optimization variable x_{count} with a domain $D_{count} = \{3,4,...,n^{G3} * m^{G3}\}$, and a set variable $x^S_{i,j}$ for every position on the scaled map. The domain of each set variable $x^S_{i,j}$ consists of the positions in the set $S_{i,j}$. When a boolean variable $x_{i,j}$ is set to true, it indicates that there is a beacon at position (i,j). If a set variable $x^S_{i,j}$ contains the value (a,b), it means that the position (i,j) is covered by a signal from a beacon at position (a,b).

A constraint ($|D^S_{i,j}| \geq 3$) is created for each set variable to ensure that every set domain has at least three values, guaranteeing that each position on the map is covered by at least three beacons. Additionally, we need to ensure that a set domain only contains a value (i,j) if there is a beacon at position (i,j), meaning the boolean variable $x_{i,j}$ is set to true. Fortunately, the *boolsChanneling* constraint, described in Section 2, precisely fulfills this requirement. Therefore, the constraint specifically requires the boolean variables representing positions in $S_{i,j}$ and the set variable $x^S_{i,j}$ as inputs. All of this collectively forms the constraint optimization problem shown in Figure 5.

We used the same optimizations to improve the solution speed of the COP as in (Löffler et al., 2022). This includes different scalings, the use of parallelization through a portfolio approach, and the addition of additional constraints.

6 RESULTS

In this section, we evaluate our new constraint-based method by comparing the maximum numbers of beacons calculated using the constraint optimization method and its solution times against the COP in (Löffler et al., 2022), and two greedy-based approaches.

The best solution for the building in (Löffler et al., 2022) we could find using our boolean variable-based constraint modeling approach, is illustrated in Figure 6. Similarly, the best solution we could find for the set variable-based constraint modeling approach is shown in Figure 7. The purple squares indicate the areas where the placement of beacons is required to ensure that at least three beacon signals cover each position on the entire building level. Due to our pessimistic approach, a purple square divided by a wall signals that the beacon can be placed on either side and still ensure triple coverage.

The solution of the boolean-based COP uses 15 beacons and the solution of the new set variable-based COP uses 18 beacons, which is five and two beacons fewer than what we needed for our best manually crafted solution. We thoroughly examined for both solutions the entire floor and confirmed that every area consistently received signals from at least three

$P = (X, D, C, f)$ with:

$$
\begin{aligned}
X =\ & \{x_{i,j} \mid \forall i \in \{1,\dots,n^3\}, j \in \{1,\dots,m^3\}\} \cup && \text{(one Boolean variable for each position } (i,j) \text{ in the grid } G3\text{)}\\
& \{x^S_{i,j} \mid \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}\} \cup && \text{(one set variable for each position } (i,j) \text{ in the } n \times m \text{ map)}\\
& \{x_{count}\} && \text{(a beacon counting variable)}\\
D =\ & \{D_{i,j} = \{0,1\} \mid \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}\} \cup && \text{(at position } (i,j) \text{ is a beacon (1) or not (0))}\\
& \{D_{count} = \{3,\dots,n^{G3} * m^{G3}\}\} \cup && \text{(maximal number of beacons is between 3 and } n*m\text{)}\\
& \{D^S_{i,j} = S_{i,j} \mid \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}\}\\
& && \text{(every position } (i,j) \text{ can be covered by all beacons with position in } S_{i,j}\text{)}\\
C =\ & \{|D^S_{i,j}| \geq 3 \mid \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}\} \cup && \text{(every position } (i,j) \text{ is covered by at least 3 beacons)}\\
& \{boolsChanneling(boolvars(S_{i,j}), x^S_{i,j}) \mid \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}\} \cup\\
& && \text{(if there is a beacon at position } (i,j) \text{ then all set variables}\\
& && \text{corresponding to a position in } S_{i,j} \text{ contain the position } (i,j)\text{)}\\
& \{count(\{x_{i,j} \mid \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}\}, x_{count}\ , 1)\} && \text{(sum up the used beacons)}\\
& minimize(x_{count})! && \text{(minimize the number of used beacons)}
\end{aligned}
$$

Figure 5: The set variable COP which represents the beacon positioning problem.

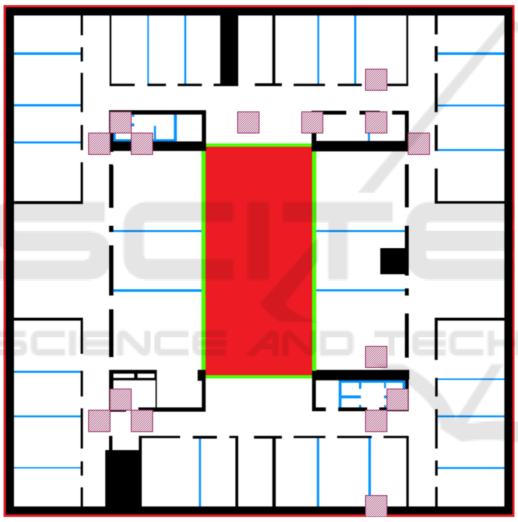
Figure 6: The best solution found with the boolean variable-based approach for the building from (Löffler et al., 2022): 15 beacons.

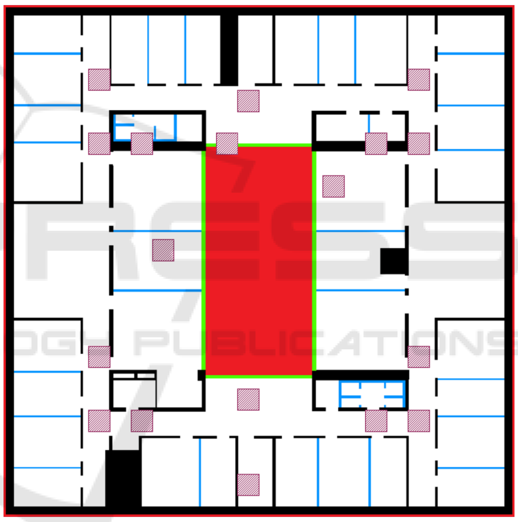
Figure 7: The best solution found with the set variable-based approach for the building from (Löffler et al., 2022): 18 beacons.

different beacons. The improvement, which is three beacons better compared to (Löffler et al., 2022), is attributed to the removal of the dynamic component in the RSSI calculation. This assumption, as mentioned before, was overly pessimistic.

To enhance the comprehension and evaluation of the effectiveness of our method, we conducted a comparative analysis with the two greedy approaches presented in (Löffler et al., 2022). The first greedy approach prioritizes covering large areas in the middle of the buildings initially and subsequently focuses on the boundaries of the buildings. Consequently, the first set of beacons covers significant portions of the building, but unfortunately, the borders are initially overlooked. This drawback results in the algorithm requiring a greater number of beacons to cover smaller areas at the borders of the building towards the end. The second algorithm assigns higher weights to hard-to-reach places such as corners, ensuring they are covered earlier. This approach results in fewer critical areas remaining where a larger number of beacons would be required at the end.

Both greedy algorithms perform significantly worse than the constraint-based approaches and yield inferior solutions. For the building in (Löffler et al., 2022), the first greedy algorithm requires 27 beacons, while the second greedy algorithm requires 28 beacons. In comparison, the constraint-based approaches outperform both greedy algorithms by achieving better results.

Table 1: A comparison of the results between the two constraint-based and greedy-based approaches.

	Boolean COP		**Set COP**		**Greedy 1**		**Greedy 2**	
	Beacons	Time	Beacons	Time	Beacons	Time	Beacons	Time
BTU Cottbus	15	207s	18	0.12s	27	0.08s	28	0.06s
Dahlewitz	9	185s	9	7.93s	11	0.01s	10	0,01s
Dresden	9	0.03s	10	0.02s	11	0.01s	12	0,01
Potsdam	43	168s	48	0.08s	50	0.48s	49	0.5s

Table 1 shows solutions with the fewest beacons found by the various approaches for different buildings, along with their corresponding solution times. It can be observed that the boolean variable-based approach from (Löffler et al., 2022) consistently finds the best solution, requiring the fewest number of beacons. The set variable-based approach, while only achieving an equally good solution for the school in Dahlewitz, quickly finds a good solution (in 3 out of 4 cases in under 1s and in one case in under 8s). The runtime of each COP was limited to 10 minutes, and the same solver settings were used. The two greedy algorithms quickly find a solution (always under 1s), but in this comparison, the solutions they find are consistently the worst.

Of course, trying our approaches for only four buildings against two simple greedy algorithms is no proof that our constraint approach always finds an optimal solution, but it can be seen as confirmation that our approach poses a valid alternative. Furthermore, it was demonstrated that our new constraint model (the set variable-based COP) is capable of solving such placement problems with good results in very short time.

The greedy approaches compute only one solution each, while the COPs, on the other hand, compute multiple solutions, from which they then select the best one, resulting in a slightly longer computation time. However, the computation times are minimal, and it can be expected or at least presumed that this scalability will hold true for larger, more complex buildings, as the beacon placements are ultimately only influenced locally due to the layout of corridors and rooms within the building.

For highly complex problem instances, two different scaling approaches can be employed. One is already inherent in our developed process, involving various scaling factors for environmental resistance A_E^s and reachability arrays A_B^S. However, this scaling approach is limited due to the restricted range of the beacons. Since the range cannot be scaled accordingly, in the more extensively scaled arrays, only a few areas are covered, resulting in significantly reduced accuracy of the overall computation and, due to the pessimistic perspective, inferior outcomes.

The alternative approach to scaling large layouts would be their division into sectors, with individual coverage calculations. At the interfaces between two sectors, there may potentially arise areas of frequent overlap, which could unnecessarily increase the overall number of required beacons. Nevertheless, it should still be possible to quickly achieve very good solutions using this method.

7 CONCLUSION AND FUTURE WORK

We introduced a new method for finding optimal solutions to a sensor beacon placement problem for indoor positioning. This method utilizes Boolean variables and set variables with Boolean channeling constraints. We conducted additional experiments to determine environmental factors for different types of walls, leading to the realization that the previously considered dynamic component was overly pessimistic and could be eliminated. We then modeled and solved a new representative Constraint Optimization Problem (COP) for the issue and outlined methods to improve the solution process. Our approach was tested on floor plans from four different buildings and compared against the approach of (Löffler et al., 2022) and two greedy algorithms. The results demonstrated the effectiveness of our method in achieving good solutions fast.

Future work involves expanding our experiments to include more buildings, conducting an experimental analysis of additional environmental factors for different wall types, incorporating additional constraints to accelerate the solution process, enhancing the signal strength estimation, and exploring the utilization of a SAT solver. Additionally, we are interested in studying the interaction of beacons across different floors and in vertical spaces such as stairwells.

REFERENCES

(1999). Typenschulbauten in den neuen ländern, sekreteriat der kultusministerkonferenz, zentralstelle für normungsfragen und wirtschaftlichkeit im bildungswesen (znwb). last visited 2023-06-20.

(2022). Global Constraint Catalog. http://sofdem.github.io/gccat/. last visited 2022-07-14.

Amsters, R., Demeester, E., Stevens, N., Lauwers, Q., and Slaets, P. (2019). Evaluation of low-cost/high-accuracy indoor positioning systems. In *The Fourth International Conference on Advances in Sensors, Actuators, Metering and Sensing, February 24 to February 28, 2019 - Athens, Greece*.

Apt, K. (2003a). *Constraint satisfaction problems: examples*. In (Apt, 2003b). Chapter 2.

Apt, K. (2003b). *Principles of Constraint Programming*. Cambridge University Press, New York, NY, USA.

Christian Schulte, Mikael Lagerkvist, G. T. (2019). Gecode 6.2.0, 2019, https://www.gecode.org/, last visited 2019-11-22.

Dechter, R. (2003a). Constraint networks. In (Dechter, 2003b), chapter 2, pages 25–49.

Dechter, R. (2003b). *Constraint processing*. Elsevier Morgan Kaufmann, San Francisco, CA 94104-3205, USA.

Fang, S., Wang, C., Huang, T., Yang, C., and Chen, Y. (2012). An enhanced zigbee indoor positioning system with an ensemble approach. *IEEE Commun. Lett.*, 16(4):564–567.

Grottke, J. and Blankenbach, J. (2021). Evolutionary optimization strategy for indoor position estimation using smartphones. *Electronics*, 10(5).

Kuchcinski, K. and Szymanek, R. (2013). Jacop - java constraint programming solver. CP Solvers: Modeling, Applications, Integration, and Standardization, co-located with the 19th International Conference on Principles and Practice of Constraint Programming ; Conference date: 16-09-2013.

Kuxdorf-Alkirata, N., Maus, G., and Brückmann, D. (2019). Efficient calibration for robust indoor localization based on low-cost BLE sensors. In Lee, H. and Geiger, R. L., editors, *62nd IEEE International Midwest Symposium on Circuits and Systems, MWSCAS 2019, August 4-7, 2019*, pages 702–705, Dallas, TX, USA. IEEE.

Löffler, S., Kroll, F., Becker, I., and Hofstedt, P. (2022). Optimal beacon placement for indoor positioning using constraint programming. In *19th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2022, December 5-8, 2022*, pages 1–8, Abu Dhabi, United Arab Emirates. IEEE.

Marriott, K. (1998). *Programming with Constraints - An Introduction*. MIT Press, Cambridge.

McGuire, J., Law, Y. W., Chahl, J., and Doğançay, K. (2021). Optimal beacon placement for self-localization using three beacon bearings. *Symmetry*, 13(1).

Perron, L. and Furnon, V. (2023). Google LLC, Google OR-Tools, 2023. https://developers.google.com/optimization/, last visited 2023-06-30.

Prud'homme, C., Fages, J.-G., and Lorca, X. (2017). Choco documentation.

Rajagopal, N., Chayapathy, S., Sinopoli, B., and Rowe, A. (2016). Beacon placement for range-based indoor localization. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8.

Rossi, F., Beek, P. v., and Walsh, T. (2006). *Handbook of Constraint Programming*. Elsevier, Amsterdam, First edition.

Sharma, R. and Badarla, V. (2018). Geometrical optimization of a novel beacon placement strategy for 3d indoor localization. In *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pages 1–6.

Subedi, S. and Pyun, J. (2020). A survey of smartphone-based indoor positioning system using rf-based wireless technologies. *Sensors*, 20(24):7230.

Taskan, A. K. and Alemdar, H. (2021). Obstruction-aware signal-loss-tolerant indoor positioning using bluetooth low energy. *Sensors*, 21(3):971.

Tomažič, S. and Škrjanc, I. (2021). An automated indoor localization system for online bluetooth signal strength modeling using visual-inertial slam. *Sensors*, 21(8).

van Hoeve, W.-J. and Katriel, I. (2006). *Global Constraints*. In (Rossi et al., 2006), First edition. Chapter 6.

Wang, H., Rajagopal, N., Rowe, A., Sinopoli, B., and Gao, J. (2019). Efficient beacon placement algorithms for time-of-flight indoor localization. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Chicago, IL, USA, November 5-8*, pages 119–128.

Zhang, D., Yang, L. T., Chen, M., Zhao, S., Guo, M., and Zhang, Y. (2016). Real-time locating systems using active RFID for internet of things. *IEEE Syst. J.*, 10(3):1226–1235.