Waterline Fault Prediction and Maintenance

Ariel Gorenstein and **Meir Kalech**
Ben-Gurion University of the Negev, Beer-Sheva, Israel

Abstract

Many water utilities employ proactive maintenance strategies in their waterline system to prevent failures. These strategies make use of a prediction model to detect the areas of the network that are likely to fail, and replace the areas with the highest probabilities. This paper suggests an alternative replacement strategy, which considers not only failure probabilities of water mains, but also other factors such as sending a team to the location and the disruption to service during the replacement, known as overhead costs. We prefer to replace waterlines which are close to each other in order to economize these overhead costs. We propose a greedy approximation algorithm to suggest an economical replacement strategy, as well as optimal and sub-optimal heuristic search algorithms. Evaluation on a real-world waterline network shows that near-optimal solutions can be effectively obtained in very fast time, which can greatly reduce the maintenance cost of the network.

1 Introduction

Waterline distribution networks experience thousands of yearly failures, such as breaks, leaks and explosions. Those failures cost a great amount of resources to the water facilities, as not only the supply lines need to be repaired, but also water is wasted and the distribution service is interrupted. For that reason many water facilities put maintenance strategies to use, which aim to replace likely-to-fail areas of the pipes in advance to prevent the failures.

Those predictive maintenance strategies often employ a prediction model that can predict the likelihood of each waterline segment to fail. Given those predictions, the facility should determine which segments should be prioritized for replacement, assuming there is a limited replacement budget. A naive replacement policy would simply replace the few segments with the highest failure probabilities.

Although such policy is likely to prevent the highest amount of failures, it might not utilize the replacement budget very effectively. The reason is that it is often more lucrative to replace adjacent waterline segments, because they can be replaced in a single replacement action. For example, suppose that replacing a segment is job that requires four workers. Hence if we want to replace the three segments with the highest failure probabilities, assuming they are in different locations, we send four workers to replace each one and have to pay for a total of 12 workers. However, if instead we replace three consecutive segments, with reasonably high failure probabilities, we would only have to send seven workers, because this replacement can be done as a single replacement action. This would leave us with more resources to replace more segments, and it is an incentive to take into account locations of segments in addition to their failure probabilities.

Several approaches have been proposed to address prioritizing waterline replacement. Kim et al. [3] model the problem as a periodic replacement problem, where the task is to find the optimal schedule for replacement and is solved using Dynamic Programming. Choi et al. [1] solve the problem using rank aggregation methods. However, no previous work takes into account the locations of waterlines or their effect on maintenance cost or policies.

Our approach aims to minimize both the cost of replacement and the expected fix cost, which is the cost of failures in segments we do not replace. We break the replace cost down into two types of costs: the material cost - which increases linearly as we replace more segments, and the overhead cost - which increases slower as we replace more adjacent segments - as explained in the above example, and prompts to replace adjacent segments. Our goal is to minimize the total maintenance cost, which is composed of the sum of the material cost, overhead cost, and expected fix cost.

We propose several approaches for that purpose. The first one is a greedy approximation algorithm - which iteratively constructs the solution until it runs of replacement budget. Although this algorithm is fast, the solution it returns is not necessarily optimal. The second approach utilizes heuristic search algorithms, specifically A*, Lookahead A*, and Weighted A*.

We evaluate the algorithms by running them on the waterline network of Israel. The results show that approximation algorithms return a solution very fast, which is very close in terms of its cost to the optimal solution. On the other hand, optimal algorithms take longer to run, and their solutions provide little improvement over the non-optimal algorithms.

2 Background and Related Work

Predictive maintenance refers to the task of preventing system malfunction by detecting components which are likely to be faulty and replacing them in advance. In the context of maintaining waterline networks, this would mean generating predictions of segments which are likely to fail, and using them to replace segments in a way that would reduce the cost of maintenance as much as possible.

Numerous methods have been suggested to prioritize replacement of watermains. In [4] five replacement strategies for waterlines in NYC were simulated: four strategies that replace segments that have had one, two three and four faults respectively, and a do-nothing strategy. Results show that the at-least-two-breaks policy is the one that yields the least costs. However, it is advised not to use the same strategy across all domains, since watermain networks with many large-diameter pipes usually benefit economically from passive policies, while those that contain a lot of small-diameter pipes usually benefit more from aggressive strategies. This is due to the fact that diameter does not increase by much the cost of fixing a pipe, but it does greatly increase the cost of replacing it.

In another study [3] the task is modeled as a Markov Decision Process (MDP) and solved optimally using a Dynamic Programming (DP) algorithm. The MDP model is constructed as follows: pipes are classified to states according to a criterion such as total failure count, and the transition probability between states is obtained either via regression or via survival analysis. The optimal solution for the replacement scheduling is then obtained with a DP, which decomposes the problem into subproblems, solves them, and uses the solutions to generate the overall result.

Choi et al. [1] model the problem as preference ordering and suggests a rank aggregation method for optimal replacement given budget constraints. This approach ranks watermains based on several criteria: reliability, replacement cost, and number of households. Then, a rank aggregation technique combines the rankings to obtain a replacement policy. This demonstrates similar performance to previous studies.

In conclusion, several studies have been proposed which recommend replacement strategies for waterlines. However, as far as we know, no paper has been published which deals with situations where the replacement cost includes a a non-linear overhead cost. This is the contribution we would like to propose in our work.

3 Problem Definition

Our goal in this paper is to propose how to utilize failure probability predictions of waterline segments in a given time range $[0, T]$ in order to suggest an economical replacement policy. In this section we formalize this task as the Expected Maintenance cOsT mInimizatiON (EMOTION) problem. Then in Section 4 we present several algorithms to solve this problem. To formalize this problem, we first define a waterline segment.

Definition 1 (**segment**). *a segment s_i is a tuple $\langle i, r_i, p_i, f_i \rangle$, where i is the index of the segment within the waterline, r_i is the material cost of replacing the segment, p_i is the probability the segment would fail in the time range $[0, T]$, and f_i is the cost of fixing the segment in case it would fail.*

The material replace cost and the fix cost of each segment are determined by the operator, and can depend on factors such as the segment's length, material, diameter, or other properties.

If there are two segments $s_i = \langle i, r_i, p_i, f_i \rangle$ and $s_j = \langle j, r_j, p_j, f_j \rangle$ such that $j = i + 1$ then s_i and s_j are considered adjacent segments. In addition, we assume that all segments belong to the same pipe. This assumption is needed to avoid situations where there are several segments that have the same indexes. We can justify this assumption in the following way: suppose we have two pipes, p_1 and p_2, where p_1 has the segments $s_{11}, s_{12}, \ldots, s_{1n}$, and p_2 has the segments $s_{21}, s_{22}, \ldots, s_{2m}$. Then, we say that the segments of p_1 have indexes $1, \ldots, n$, and the segments of p_2 have the indexes $n + 2, n + 3, \ldots, n + m + 1$.

We consider cases where the cost of replacing segments includes a **submodular overhead cost** in addition to raw material cost. This overhead could be, for instance, due to the cost of sending a team to the location and the cost of disruption to service. Every set of consecutive segments to be replaced incurs an overhead cost, and the cost increases with the number of consecutive segments. However, it increases in a concave way; that is, the more consecutive segments we replace, the less the cost increases. Intuitively, suppose replacing 100 meters of a waterline requires sending a truck and three workers. Then replacing 100 meters in three different locations would result in a total of three trucks and nine workers. However, replacing 300 consecutive meters of a waterline only requires two trucks and five workers. This gives us incentive to replace adjacent segments. The function that defines the overhead cost of replacing n consecutive segments, $o(n)$, is increasing and concave: its first derivative is positive and its second derivative is negative.

Definition 2 (**consecutive segments**). *we call a set of segments S consecutive, if for each $s_i, s_j \in S$, where $s_i = \langle i, r_i, p_i, f_i \rangle$ and $s_j = \langle j, r_j, p_j, f_j \rangle$, the following holds: $|i - j| < |S|$.*

For example, the set $S = \{s_2, s_3, s_4\}$ is consecutive, but the set $S = \{s_2, s_3, s_5\}$ is not consecutive, because $5 - 2 = 3 \not\leq 3 = |S|$.

We call $CONS$ the consecutive partitioning function, which partitions a set of segments into consecutive sets. Formally,

Definition 3 (**consecutive partitioning function**). *$CONS(S)$ is a partition of S that satisfies $\forall S' \in CONS(S)$, S' is consecutive: $CONS : 2^S \to 2^{2^S}$.*

For example, $CONS(\{s_2, s_3, s_5\}) = \{\{s_2, s_3\}, \{s_5\}\}$. Armed with this function we can formally define the overhead cost.

Definition 4 (**overhead cost**). *the overhead cost of replacing n consecutive segments is given as $o(n) = \frac{n}{n+a} \cdot h$, where a and h are positive parameters. h defines the amplitude of the function, and a determines its gradient.*

For example, suppose a case where $a = 3$ and $h = 10$. Then we get the function $o(n) = \frac{n}{n+3} \cdot 10$. So replacing one segment would cost $o(1) = \frac{1}{4} \cdot 10 = 2.5$, and therefore replacing three segments in three different locations would cost 7.5. However, replacing three adjacent segments costs only $o(3) = \frac{3}{6} \cdot 10 = 5$. We can clearly see the gain in this case.

By calculating the first and second derivatives of $o(n)$, it can be shown that this function is increasing and concave. Note that this function could be changed by the operator to any other increasing and concave function, and all the algorithms and theorems in this paper would be correct for that function too.

The overhead cost of replacing a set of (not necessarily consecutive) segments S can be computed by using $CONS$ to partition S to sets of consecutive segments, calculating the overhead cost of each one, and taking the sum. We denote O for the overhead cost function of a set of segments:

$$O(S) = \sum_{S' \in CONS(S)} o(|S'|) = \sum_{S' \in CONS(S)} \frac{|S'|}{|S'| + a} \cdot h$$

Note that we use the function $o(n)$ (lowercase o) for the overhead cost of replacing n consecutive segments, and $O(S)$ (uppercase O) for the overhead cost of replacing a set of segments S.

Definition 5 (**replacement cost**). *the replacement cost of a set of segments S_r, denoted as $R(S_r)$, is the sum of the material replace costs of the segments in S_r with the overhead cost of S_r. Formally, $R(S_r) = \sum_{s_i \in S_r} r_i + O(S_r)$.*

We assume we are given a replacement budget B that constrains the amount we can spend on replacement. Given the set of segments S and the replacement budget B, our algorithms should propose which subset of segments S_r to replace such that $R(S_r) \leq B$.

The expected fix cost in the time range $[0, T]$ of replacing the segments $S_r \subseteq S$ is the cost of fixing the segments not in S_r, and it depends on the failure probabilities of those segments, p_i and on their fix costs, f_i.

Definition 6 (**expected fix cost**). *given a set of segments S, the expected fix cost of replacing the subset $S_r \subseteq S$, in the time range $[0, T]$, is given as $F(S, S_r) = \sum_{s_i \in S \setminus S_r} p_i \cdot f_i$.*

Finally, the expected maintenance cost, which is the target function we aim to minimize, is the sum of the replacement cost and the expected fix cost.

Definition 7 (**expected maintenance cost**). *the expected maintenance cost of replacing a set of segments S_r out of a set S in the time range of $[0, T]$ is the combined cost of the replacement cost of S_r with the expected fix cost of S and S_r:*

$$M(S, S_r) = R(S_r) + F(S, S_r)$$

If we use the previous definitions we get:

$$M(S, S_r) = \sum_{s_i \in S_r} r_i + \sum_{S' \in CONS(S_r)} \frac{|S'| \cdot h}{|S'| + a} + \sum_{s_i \in S \setminus S_r} p_i \cdot f_i$$

Definition 8 (**Expected Maintenance cOsT mInimizatiON Problem (EMOTION)**). *given a set of segments S and the replacement budget B, EMOTION problem aims to compute $S_r \subseteq S$ such that $M(S, S_r) = R(S_r) + F(S, S_r)$ is minimal, and $R(S_r) \leq B$.*

Example: consider an EMOTION where $S = \{s_1, s_2, s_3, s_4\}$ and $B = 60$. The failure probabilities of the segments are $0.3, 0.05, 0.25, 0.25$ respectively. All four segments have material replace cost r_i of 15 and fix cost f_i of 100. Also, $a = 1$ and $h = 30$. Suppose we choose to replace s_1 and s_2. That is, $S_r = \{s_1, s_2\}$. The raw material cost of this replacement is 30, since both s_1 and s_2 have material cost of 15. The overhead cost would be $o(2) = \frac{2}{2+1} \cdot 30 = 20$, since the two segments being replaced are adjacent. Therefore the total replacement cost is 50, which is within the budget constrains. The expected fix cost is the sum of the failure probabilities of s_3, s_4 multiplied by their fix cost: So $F(S, S_r) = p_3 \cdot f_3 + p_4 \cdot f_4 = 0.25 \cdot 100 + 0.25 \cdot 100 = 50$. Finally, the expected maintenance cost is the sum of replacement and expected fix costs: $50 + 50 = 100$.

4 Algorithms

We propose several algorithms to solve this optimization problem. The first one is a greedy approximation algorithm (Section 4.1), and the second set includes heuristics search based algorithms (Section 4.2).

4.1 Greedy Algorithm

This is a greedy approximation algorithm, which constructs the solution by adding a single segment at a time. Each iteration, the segment that improves the expected maintenance cost the most is added. The algorithm stops when the replacement budget runs out, or when it reaches a state where adding more segments to the solution does not benefit the expected maintenance cost.

Although the solution returned by this algorithm is not necessarily optimal, its advantage is its polynomial runtime. During each iteration of the algorithm we have to run through all segments not yet in the solution, and for each one compute the difference in the expected maintenance cost between the solution without the segment and the solution with the addition of the segment. This takes $O(n)$ time, where n is the number of segments in the network. The total number of iterations can be bounded by n, since one segment is added to the solution at each iteration. Hence the total runtime of the algorithm is $O(n^2)$.

4.2 Heuristic Search Algorithms

By modeling the task of selecting the segments for replacement as a search problem, we can solve it optimally using generic heuristic search algorithms. In order to do so, we define how **states** are represented, the **transitions** between states, the **start and goal** states, and the **heuristic** function.

First we define a function $T\colon 2^S \longrightarrow 2^{2^S}$ called the **transition function**. For a set of segments $S_r \in 2^S$, let $s_i \in S_r$ be the segment in S_r with the highest index. For $s_j \notin S_r$, denote $S_r + s_j = S_r \cup \{s_j\}$. Then $T(S_r) = \{S_r + s_j \mid j > i \wedge R(S_r + s_j) \leq B \wedge M(S, S_r + s_j) < M(S, S_r)\}$. That is, $T(S_r)$ are the

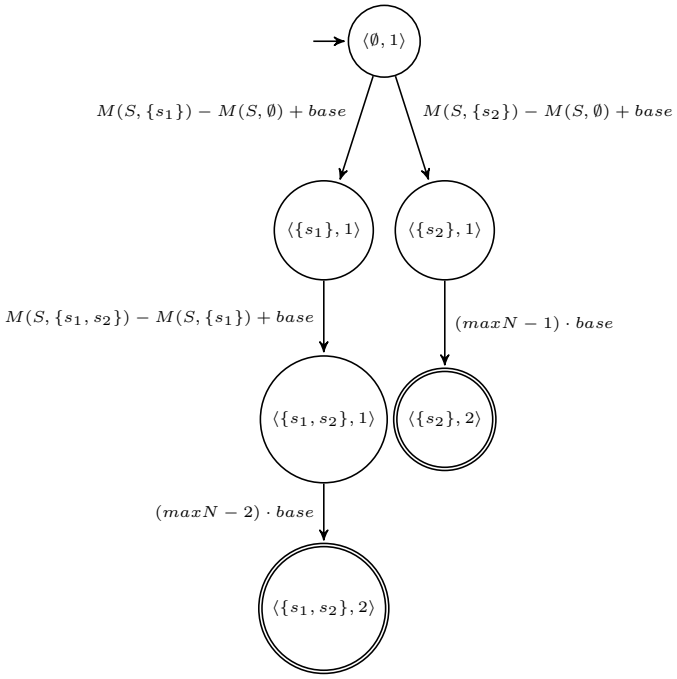

Figure 1: Example set space

subsets of segments that result in adding a segment s_j to S_r, which satisfy that the index of s_j is larger than the maximal index in S_r, and that this addition is affordable with the budget, and that the addition reduces the expected maintenance cost.

Next, we define $S_F = \{S_r \subseteq S \mid T(S_r) = \emptyset\}$. That is, S_F contains the subsets of segments to which no segments with a higher index can be added in a way that would be affordable and that would reduce the expected maintenance cost. Subsets in S_F can be called "final subsets", since no further segments can be added to them.

Given S_F, we define the **state space** as follows: for every set of segments S_r, there is a state of the form $\langle S_r, 1\rangle$. In addition, if S_r is in S_F, there is also a state $\langle S_r, 2\rangle$. This means that each subset of segments in S_F gets two states. Formally, the state space is $(2^S \times \{1\}) \cup (S_F \times \{2\})$. The **start state** is $\langle\emptyset, 1\rangle$, and for all $S' \in S_F$, $\langle S', 2\rangle$ is a **goal state**.

For example, suppose $S = \{s_1, s_2\}$ and that $T(\emptyset) = \{\{s_1\}, \{s_2\}\}$, $T(\{s_1\}) = \{\{s_1, s_2\}\}$, $T(\{s_2\}) = \emptyset$, $T(\{s_1, s_2\}) = \emptyset$. Then $S_F = \{\{s_2\}, \{s_1, s_2\}\}$. Figure 1 shows the state space in that case. Each circle represents a state, and double circles are goal states. $\langle\emptyset, 1\rangle$ is the start state.

Next we define transitions between states. Let $\langle S_r, k\rangle$ be a state.

- If $k = 2$ then $\langle S_r, k\rangle$ is a goal state, so there are no transitions from it.
- If $k = 1$ and $S_r \notin S_F$: for each $S'_r \in T(S_r)$, there is a transition from $\langle S_r, 1\rangle$ to $\langle S'_r, 1\rangle$. The cost of the transition is $M(S, S'_r) - M(S, S_r) + base$, given that $base = p_{max} \cdot f_{max}$, where p_{max} is the highest failure probability among all segments and f_{max} is the highest fix cost among all segments. Since by the definition of $T(S_r)$, $M(S, S'_r) < M(S, S')$, the addition of $base$ is needed to ensure positive transition costs. When adding a segment the expected maintenance cost can decrease by at most $p_{max} \cdot f_{max}$, and therefore adding $base$ guarantees that transition costs are not negative.
- If $k = 1$ and $S_r \in S_F$: there is a single transition from $\langle S_r, 1\rangle$ to $\langle S_r, 2\rangle$ (which is a goal state). The cost of the transition is $(maxN - |S_r|) \cdot base$, where $maxN$ is an upper bound on the number of segments that any solution can possibly have. It can be calculated as the largest value of n such that $r_{min} \cdot n + \frac{n}{n+a} \cdot h \leq B$ (where r_{min} is the lowest material replace cost among all segments). Later, we explain the reason that we cannot define $\langle S_r, 1\rangle$ as the goal state and instead add another transition from it to $\langle S_r, 2\rangle$.

The arrows in figure 1 display the transitions in the above example, along with their costs.

A solution to the problem is a sequence of states $\langle S_{r_0}, 1\rangle, \langle S_{r_1}, 1\rangle, \langle S_{r_2}, 1\rangle, \ldots, \langle S_{r_m}, 1\rangle, \langle S_{r_m}, 2\rangle$, which hold that $S_{r_0} = \emptyset$, $S_{r_m} \in S_F$ and for $i = 1, \ldots, m$: $S_{r_i} \in T(S_{r_{i-1}})$.

The cost of the last transition is $(maxN - |S_{r_m}|) \cdot base = (maxN - m) \cdot base$. The sum of costs of all the other transitions is $\sum_{i=1}^{m} [M(S, S_{r_i}) - M(S, S_{r_{i-1}}) + base] = \sum_{i=1}^{m} [M(S, S_{r_i}) - M(S, S_{r_{i-1}})] + m \cdot base$. Since this expression includes a telescopic sum, it can be simplified to $M(S, S_{r_m}) - M(S, S_{r_0}) + m \cdot base = M(S, S_{r_m}) - M(S, \emptyset) + m \cdot base$. When we add the cost of the final transition we get that the total cost is $M(S, S_{r_m}) - M(S, \emptyset) + maxN \cdot base$. The only non-constant in this expression is $M(S, S_{r_m})$. Therefore a solution with minimal cost ensures that the resulting S_{r_m} would be of minimal expected maintenance cost.

The reason that we cannot define $\langle S_{r_m}, 1\rangle$ as the goal state and instead add another transition from it to $\langle S_{r_m}, 2\rangle$ is to ensure the the solution is optimal. If we remove the last edge, the cost of the solution would be $M(S, S_{r_m}) - M(S, \emptyset) + m \cdot base$. Since this value is dependent on m, it may cause the search algorithm to favor short solutions over long ones, and return a solution that is not optimal in terms of expected maintenance cost. For that reason we add another edge of cost $(maxN - |S_{r_m}|) \cdot base$ to remove this dependency.

The only thing left to define is the heuristic function. Given a state $\langle S_r, k\rangle$, this function should underestimate the cost from this state to any goal state.

- If $k = 2$, then this a goal state and its heuristic value is 0.
- If $k = 1$ and $S_r \in S_F$, we know the cost to the goal state exactly: $(maxN - |S_r|) \cdot base$.
- If $k = 1$ and $S_r \notin S_F$: Let $\langle S_{r'}, 2\rangle$ be the closest goal state. The cost to get to this state is $M(S, S_{r'}) - M(S, S_r) + base \cdot (maxN - |S_r|)$. Therefore we need to find a lower bound on $M(S, S_{r'}) = R(S_{r'}) + F(S, S_{r'})$. We will bound each of those two parts separately. Bounding $R(S_{r'})$ is easy: we know it's at least $R(S_r)$. In order to bound $F(S, S_{r'})$ we can find an upper bound on the additional number of segments that can added to the solution, and adding that many segments with the highest failure probabilities and fix costs. Let b

be the remaining budget: $b = B - R(S_r)$. Let s_i be the segment in S_r with the highest index. The way we can add the largest possible amount of segments to S_r without exceeding the budget is by adding segments adjacent to s_i and assuming their replace cost is the lowest among all segments. Let $C_i \in CONS(S_r)$ be the consecutive set that includes s_i, and denote $c = |C_i|$. If we add n segments adjacent to s_i, then the additional cost we pay is at least $r_{min} \cdot n + o(c+n) - o(c)$. We select the largest n such that this value does not exceed b. Afterwards, we add to S_r the n segments with indexes larger than i that have the highest $p_i \cdot f_i$. Let S_r^+ be the resulting set. Then we know that $F(S, S_r^+) \leq F(S, S_{r'})$.

Now that the search problem is defined, we can solve it with heuristic search algorithms. The algorithms we use are A* [2], Lookahead A* [6], and Weighted A* [5].

A*: The algorithm maintains an open list, initialized with the start state, and a closed list, initialized as the empty set. Every iteration, the algorithm extracts from the open list the node n with the minimal $f(n) = g(n) + h(n)$ value, where $g(n)$ is the cost of the best path found so far from the start state to n, and $h(n)$ is the heuristic value of n. n is added to the closed list and all of its children are added to the open list. When a goal state is extracted from the open list, the algorithm stops, and that state is guaranteed to be the optimal solution.

Lookahead A* (LA*): This is an optimal algorithm based on A*, with the main difference being that every time a state n is chosen for expansion, a depth first search is performed from n down to a threshold of $f(n)+k$, where k is a predefined constant. The motivation is that if a goal state is found during the depth first search, the search might be able to terminate sooner. We chose to use 3 for the value of k.

Weighted A* (WA*): This algorithm is identical to A*, except that it boosts the heuristic value of all states by a constant w. This increase helps the algorithm find the solution faster. The downside is that this may cause the heuristic to become not admissible which results in a non optimal solution. However, the cost of the returned solution is guaranteed to be at most w times the cost of the optimal solution [5]. We used a small value of $w = 1.0625$. This was enough to increase the speed of the algorithm significantly without much loss to the quality of the solution.

5 Evaluation

In this section we present experimental evaluation to examine the properties of the suggested methods. In particular we would like to address the following research questions:

RQ1. *How parameters such as replace cost, overhead cost and fix cost affect the gaps in runtime between the algorithms?*
RQ2. *How parameters such as replace cost, overhead cost and fix cost affect the gaps in solution cost between the algorithms?*
RQ3. *Which of the optimal search algorithm has the best performance?*

5.1 Experiment Setup

We used the water network of Israel which contains 417 pipes with a total length of 1747 km. The overall fault count of all pipes during a 39 month period is 2095. We divided the pipes into segments of 400 meters each, to a total of 4577 segments. Throughout the experiments, we assumed that all segments have a uniform material replace cost r and a uniform fix cost f: $\forall s_i \in S$: $r_i = r$ and $f_i = f$.

We used the Random Forest Regression algorithm to predict the failure probability of each segment. We did this by dividing the faults to three equal periods of 13 months each, that had 563, 797 and 735 faults respectively. We trained the algorithm using the segments' features, which include their physical characteristics such as age and material, as well as their fault count in the first period. We fit those features to the fault counts in the second period. After the algorithm was trained, we inputted the segment features and faults in the second period to predict faults in the third period. Finally, we normalized the results to the range of $[0, 1]$ so they predict failure probabilities rather than failure counts. Given those predictions of failure probabilities in the third period, our algorithms decide at the end of the second period, which segments to replace.

We made an experiment for each of the following four parameters: the number of segments in the network (Segment Count), the material replace cost of the segments (r), the overhead coefficient (h), and the fix cost of the segments (f). During the experiment of each parameter, we altered the value of that parameter while fixating the others: segment count - 30, r - 12, h - 62, f - 3000. We solved 20 random instances of each setting using the following algorithms: Greedy, A*, LA*, WA* and Baseline. Baseline is a simple algorithm that sorts the segments by their failure probabilities in descending order and replaces segments from the top of the resulting list until the budget runs out. We measured the average time it took each algorithm to solve those 20 instances, as well as the average expected maintenance cost.

5.2 Results

Results of the experiment are shown in figures 2-5. Each of the figures corresponds to one of the four parameters: segment count, replace cost (r), overhead cost (h) and fix cost (f). The figure that corresponds to each parameter describes the experiments where that parameter was altered, while the rest of the parameters were fixated. The left side of each figure displays the expected maintenance cost results, and the right side displays the runtime results. The results show a graph for each of the algorithms: Greedy, A*, LA*, WA* and Baseline.

Since A* and LA* are both optimal algorithms and the solutions of both always have the exact same expected maintenance cost, their graphs in the expected maintenance cost results overlap. Also, Baseline overlaps with Greedy in the time results, because both have very similar runtime.

In Figure 2 we can see the expected maintenance cost results and the runtime results of the algorithms for segment counts between 30 and 60. Naturally, both

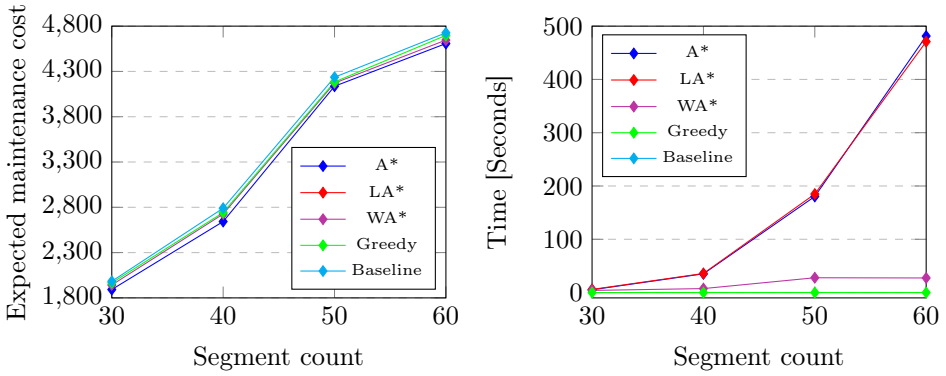

Figure 2: Segment count results.

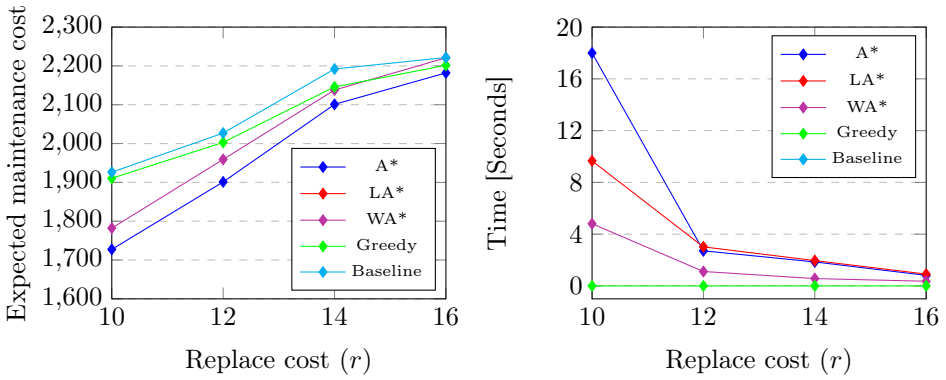

Figure 3: Replace cost results.

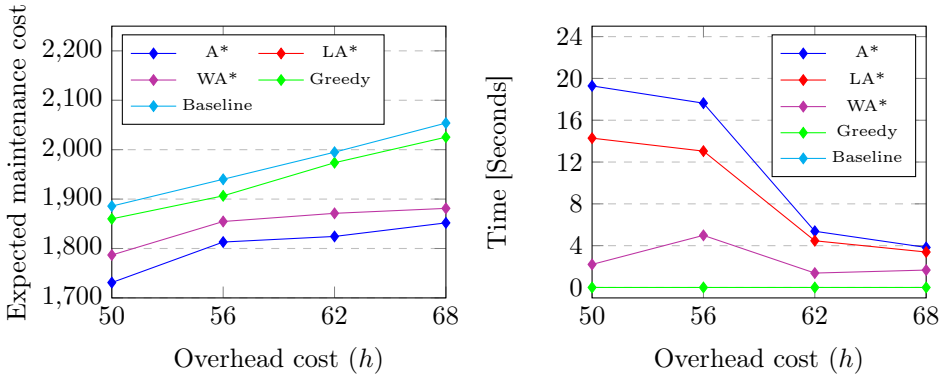

Figure 4: Overhead cost results.

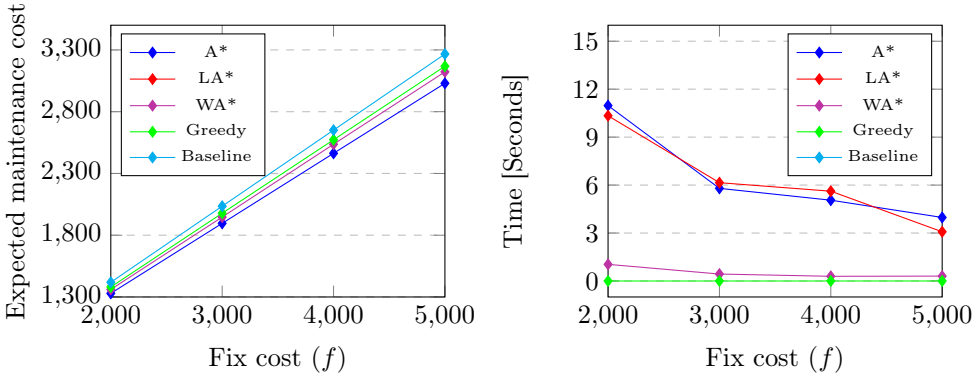

Figure 5: Fix cost results.

the solution cost and the runtime of all algorithms increase as the number of segments in the network increases. However, it is clear that the runtime of A* and LA* increases much faster than the runtime of WA* or Greedy, which are much more scalable. Also, it is evident that although the approximation algorithms return non-optimal solutions, the cost of their solutions is very close to the optimal cost.

Figure 3 corresponds to the material replace cost of the segments. As for the runtime, which decreases as the cost increases, the reason is natural: when the cost of replacing segments becomes high, the replacement budget allows to replace less segments, and hence the depth of the search tree decreases. As for the expected maintenance cost, we can see that the lower the material replace cost - the greater the gap is between greedy and A*. The reason is that with lower material replace cost, the budget allows to replace more segments, which gives the algorithms more solution options and therefore more room for error for non-optimal algorithms.

In Figure 4 that displays the results for the overhead cost, the runtime results show similar trends as in the previous case: larger costs imply lower time, due to the reduction of the search tree depth. The expected maintenance cost results show that the gap between the greedy and optimal costs increases with the overhead cost. The reason is that when the overhead cost is high, it is more meaningful to consider locations of segments. Optimal solutions in this case often include segments which do not have very high failure probabilities, which makes simple algorithms that mainly consider the probabilities more prompt to make a mistake.

Finally, Figure 5 shows the results for the fix cost parameter. As the cost increases, the gaps between the expected maintenance costs also slightly increase in favor of the optimal algorithms. With high fix costs, algorithms that make a mistake and do not replace the correct segments, pay more for failures. Concerning the runtime results, we can see that higher fix cost means faster runtime. This is due to the fact that when the fix cost becomes significantly higher than the overhead cost, there is less importance to the locations of the segments being replaced, and more importance to their probabilities, which makes finding the solution easier.

We can conclude that:

- Greedy is a very fast algorithm which usually returns solutions with costs that are very close to the optimal cost, and it always outperforms Baseline. Regarding RQ2, the gap between the greedy solution cost and the optimal solution cost increases as the overhead cost and fix cost increase, and it decreases as the material replace cost increases.
- WA* is by far the fastest heuristic search algorithm, and the solution it returns is better than that of greedy in almost all cases. Unlike the other search algorithms, its runtime increases very slowly with the network size, and can therefore be used on very large networks. Regarding RQ1, the runtime gap between WA* and the other search-based algorithm decreases as the material replace cost, overhead cost, and fix cost increase.
- To answer RQ3, LA* is consistently faster than A*, so we can conclude that enhancing A* with lookaheads is beneficial to the runtime. Since adding lookaheads does not chance the optimality of the solution, LA* is better to use than A*.
- Although A* and LA* return optimal solutions, their slow runtime makes them impractical on large networks. In cases where finding the exact optimal solution is not mandatory, it is better to use faster algorithms such as WA* or greedy, which return near-optimal solutions.

6 Conclusions and Future Work

We addressed the importance of proactive maintenance in waterline networks and the need to replace certain segments in advance. We presented the task of maintenance in cases where the replacement cost consists both of raw material costs and of an additional submodlar overhead cost that depends on adjacency of replaced segments. We proposed several algorithms to prioritize segment replacement, including a greedy algorithm and heuristic search algorithms. Experimental results show that it is important to consider adjacency of segments during replacement and not only failure probabilities, and often it is more lucrative to replace adjacent segments even if they do not have the highest probabilities.

Future work will focus on developing an additional algorithm that utilizes the submodularity property of the expected maintenance cost function. There are existing polynomial time algorithms that can optimally minimize submodular set functions without constrains. Since we have the budget constraint we cannot use those algorithms directly, but we would like to modify them to propose an approximate algorithm. In addition, we intend to theoretically analyze the greedy algorithm, in order to find bounds on its solution cost, and identify properties of networks on which it returns a non-optimal solution.

References

[1] Go Bong Choi et al. "A prioritization method for replacement of water mains using rank aggregation". In: *Korean Journal of Chemical Engineering* 34.10 (2017), pp. 2584–2590.

[2] Peter E Hart, Nils J Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[3] Jong Woo Kim et al. "Dynamic optimization of maintenance and improvement planning for water main system: Periodic replacement approach". In: *Korean Journal of Chemical Engineering* 33.1 (2016), pp. 25–32.

[4] James W Male, Thomas M Walski, and Adam H Slutsky. "Analyzing water main replacement policies". In: *Journal of Water Resources Planning and Management* 116.3 (1990), pp. 362–374.

[5] Ira Pohl. "Heuristic search viewed as path finding in a graph". In: *Artificial intelligence* 1.3-4 (1970), pp. 193–204.

[6] Roni Tzvi Stern et al. "Using lookaheads with optimal best-first search". In: *Twenty-Fourth AAAI Conference on Artificial Intelligence*. 2010.