Bamboo Garden Trimming Problem* (Perpetual maintenance of machines with different attendance urgency factors)

Leszek Gąsieniec Ralf Klasing Christos Levcopoulos
Andrzej Lingas Jie Min Tomasz Radzik

Abstract

A garden G is populated by $n \geq 1$ bamboos $b_1, b_2, ..., b_n$ with the respective daily growth rates $h_1 \geq h_2 \geq \cdots \geq h_n$. It is assumed that the initial heights of bamboos are zero. The robotic gardener or simply a robot maintaining the bamboo garden is attending bamboos and trimming them to height zero according to some schedule. The *Bamboo Garden Trimming Problem*, or simply BGT, is to design a perpetual schedule of cuts to maintain the elevation of bamboo garden as low as possible. The bamboo garden is a metaphor for a collection of machines which have to be serviced with different frequencies, by a robot which can service only one machine during a visit. The objective is to design a perpetual schedule of servicing the machines which minimizes the maximum (weighted) waiting time for servicing.

We consider two variants of BGT. In *discrete* BGT the robot is allowed to trim only one bamboo at the end of each day. In *continuous* BGT the bamboos can be cut at any time, however, the robot needs time to move from one bamboo to the next one and this time is defined by a weighted network of connections.

For discrete BGT, we show a simple 4-approximation algorithm and, by exploiting relationship between BGT and the classical Pinwheel scheduling problem, we obtain also a 2-approximation and even a closer approximation for more balanced growth rates. For continuous BGT, we propose approximation algorithms which achieve approximation ratios $O(\log(h_1/h_n))$ and $O(\log n)$.

1 Introduction

In this paper we consider a perpetual scheduling problem in which a collection of (possibly virtual) machines need to be attended with very often *known* but possibly different frequencies, i.e., some machines need to be attended more often than others. We model such scheduling problems as *Bamboo Garden Trimming (BGT) Problem*. A collection (garden) G of n bamboos b_1, b_2, ..., b_n with known respective daily growth rates

*L. Gąsieniec and J. Min's work was partially supported by Network Sciences and Technologies (NeST) at University of Liverpool. R. Klasing's work was partially funded by the ANR project DISPLEXITY (ANR-11-BS02-014). This study has been carried out in the frame of "the Investments for the future" Programme IdEx Bordeaux – CPU (ANR-10-IDEX-03-02). T. Radzik's work was supported in part by EPSRC grant EP/M005038/1, "Randomized algorithms for computer networks."

$h_1 \geq h_2 \geq \cdots \geq h_n > 0$ is given. Initially the height of each bamboo is set to zero. The robotic gardener maintaining the garden trims bamboos to height zero according to some schedule. The height of a bamboo b_i after $t \geq 0$ days is equal to $(t - t')h_i$, where t' is the last time when this bamboo was trimmed, or $t' = 0$, if it has never been trimmed by time t. The main task in BGT is to design a perpetual schedule of cuts to keep the highest bamboo in the garden as low as possible, while complying with some specified constraints on the timing of cutting. The basic constraints considered in this paper are that the gardener can cut only one (arbitrary) bamboo at the end of each day and is not allowed to attend the garden at any other times. Once the gardener has decided which bamboo to trim in the current round (at the end of the current day), the action of actual trimming is instantaneous. The problem, while of inherent combinatorial interest, originates from perpetual testing of virtual machines in cloud systems [1]. In such systems frequency in which virtual machines are tested for undesirable symptoms vary depending on importance of dedicated cloud operational mechanisms.

BGT is also a natural extension of several classical algorithmic problems with the focus on *monitoring* and *mobility*, including the *Art Gallery Problem* [17] and its dynamic extension called the *k-Watchmen Problem* [20]. In a more recent work on *fence patrolling* [9, 10] the studies focus on monitoring vital (possibly disconnected) parts of a linear environment where each point is expected to be attended with the same frequency. The authors of [11] study monitoring linear environments by robots prone to faults. Our paper focuses on the case where each vital part of the environment has its own, possibly unique urgency factor, which makes it related to *periodic scheduling* [19], a series of papers on the *Pinwheel* problems [6, 7, 13] including the *periodic Pinwheel* problem [14, 16] and the *Pinwheel scheduling* problem [18], as well as the concept of *P-fairness* in sharing multiple copies of some resource among various tasks [2, 3].

We consider two variants of the BGT problem. The constraints that only one bamboo is cut at the end of each day define *discrete* BGT. In the second variant, *continuous* BGT, we assume that for any two bamboos b_i and b_j, we know the time $t_{i,j} > 0$ that the robot needs to relocate from b_i to b_j. In this variant the time when the next bamboo is trimmed depends on how far that bamboo is from the bamboo which has just been trimmed. As in discrete BGT, when the robot arrives at the bamboo which is to be trimmed, the actual action of trimming is instantaneous. We assume that the travel times are symmetric, that is, $t_{i,j} = t_{j,i}$, and can be fractional. Previous work on problems of similar nature as the continuous BGT includes recent work on patrolling [9, 10, 11, 15].

In related research on minimizing the maximum occupancy of a buffer in a system of n buffers, the usual setting is a game between the player and the adversary [4, 5, 8]. The adversary decides how the fixed total increase of data in each round is distributed among the buffers and tries to maximize the maximum occupancy of a buffer. The player decides which buffer (or buffers, depending on the variant of the problem) should be emptied next and tries to minimize the maximum buffer size. The upper bounds developed in this more general context can be translated into upper bounds for our BGT problems, but our aim is to derive tighter bounds for the case when the rates of growth of bamboos are fixed and known.

Probably the most natural strategy to keep the elevation of the bamboo garden low is the greedy approach of always cutting next the highest bamboo. This approach, called *Reduce-Max*, was considered recently in the context of periodic testing of virtual machines in cloud systems [1], and was also studied in the adversarial setting of the

buffer minimization problems mentioned above. The results presented in [5] imply a tight bound of $H(H_{n-1}+1) = \Theta(H \log n)$ on the performance of Reduce-Max for discrete BGT when the adversary keeps changing the growth rates of bamboos, where H is the sum of the daily growth rates (the adversary cannot change this sum) and $H_k = \sum_{i=1}^{k} \frac{1}{k} = \Theta(\log k)$ is the k-th harmonic number. While the $O(H \log n)$ upper bound applies obviously also to our setting of the discrete BGT, when the growth rates are fixed, it is not clear whether there are instances which force Reduce-Max to leave bamboos of height $\Omega(H \log n)$. On the contrary, the experimental work presented in [1] indicates possibility that Reduce-Max keeps the maximum bamboo height within $O(H)$. The upper bound of $O(H \log n)$ on Reduce-Max for discrete BGT implies an $O(DH \log n)$ upper bound on the same approach for continuous BGT, where D is the diameter of the set of bamboos (the largest travel time between any pair of bamboos), but again this upper bound from the adversarial setting does not help us in analyzing how well we can do for given growth rates.

In both cases, discrete and continuous, we consider algorithms $\mathcal{A}$ which for an input instance I (of the form $\langle h_i : 1 \leq i \leq n \rangle$ in the discrete case and $[\langle h_i : 1 \leq i \leq n \rangle, \langle t_{i,j} : 1 \leq i, j \leq n \rangle]$ in the continuous case) produce a perpetual (trimming) schedule $\mathcal{A}(I)$, that is, a sequence of indices of bamboos $(i_1, i_2, \ldots)$ which defines the order in which the bamboos are trimmed. We are mainly interested in the *approximation ratios* of such algorithms, which are defined in the usual way. For an input instance I and a trimming schedule $\mathcal{S}$ for I, let $MH(\mathcal{S})$ denote the supremum of the heights of bamboos over all times $t \geq 0$ when the trimming proceeds according to schedule $\mathcal{S}$, and let $OPT(I)$ denote the infimum of $MH(\mathcal{S})$ over all schedules $\mathcal{S}$ for I. The upper bounds on Reduce-Max imply that $OPT(I)$ is finite. The approximation ratio of a schedule $\mathcal{S}$ is defined as $MH(\mathcal{S})/OPT(I)$ and the approximation ratio of an algorithm $\mathcal{A}$ is the supremum of $MH(\mathcal{A}(I))/OPT(I)$ over all input instances I. While our main goal is a low approximation ratio, we are also interested in the time complexity of BGT algorithms and try to keep low both the time of any preprocessing and the time needed to compute the index of the next bamboo in the schedule.

For each instance of discrete BGT with the sum of the growth rates $H = h_1 + h_2 + \cdots + h_n$, $OPT(I) \geq H$, as shown below. Thus the approximation ratio of Reduce-Max is $O(\log n)$ but it remains an open questions whether this upper bound is tight. In Section 2, we show that a simple modification of Reduce-Max has the approximation ratio at most 4. We also show more complicated algorithms, which are based on the relation between discrete BGT and the Pinwheel problem and have approximation ratios of 2, for any growth rate sequence, and $(1+\delta)$, for a constant $0 < \delta < 1$ and "balanced" growth rate sequences.

In Section 3, we show algorithms for continuous BGT with approximation ratios $O(\log(h_1/h_n))$ and $O(\log n)$. In the full version of our paper, we show also some hard instances of the continuous BGT problem such that for any schedule the maximum bamboo height is greater than our lower bounds by a $\Theta(\log n)$ factor. Thus for these input instances our $O(\log n)$-approximation algorithm computes in fact constant-approximation schedules. We also leave to the full version of the paper a $O(1)$-approximation algorithm for continuos BGT for the case when $h_1 = \Theta(H)$.

Lower bound on discrete BGT. We note a natural lower bound of $H = h_1 + h_2 + \cdots + h_n$ on the maximum height of a bamboo in the discrete BGT problem. Thus neither

Reduce-Max nor any other algorithm for the discrete BGT problem can keep the bamboos within the height H, that is, $\max\{MH(\mathcal{A}(I) : \text{sum of growth rates in } I \text{ is } H\}/H$ is an upper bound on the approximation ratio of an algorithm $\mathcal{A}$. This bound can be proved by contradiction. Assume there exists a perpetual schedule that keeps the heights of all bamboos below $H_{MAX} < H$. During each day the *total height* T of the bamboos, that is, the sum of the current heights of all bamboos, increases at least by $H - H_{MAX} > 0$. Thus after $\lfloor nH_{MAX}/(H - H_{MAX}) \rfloor + 1$ days the height of at least one bamboo is greater than H_{MAX} – a contradiction. A similar lower bound argument can be obtained via density restrictions in *Pinwheel* problem, discussed later in Section 2.2.1.

2 Discrete BGT

We consider two types of algorithms for the discrete variant of BGT. An *online* algorithm is based on simple queries of type "what is the tallest bamboo?" (as in Reduce-Max), or "what is the fastest growing bamboo with the height above some threshold?" (as below in Reduce-Fastest). Such queries can be answered without knowing the whole distribution of growth rates. Online scheduling is more flexible since its performance can adapt, if the growth rates change. On the downside, the performance analysis of online scheduling is more complex and the approximation bounds tend to be weaker. In contrast, an *offline* algorithm determines which bamboo is to be trimmed during a particular round by producing, based on the knowledge of the whole distribution of growth rates, the full perpetual schedule during preprocessing. This reduces the flexibility of the solution, but leads to stronger approximation bounds. We note that our online-offline characterization is to indicate only a general nature of possible BGT algorithms.

2.1 Constant approximation of BGT by online scheduling

We obtain our first constant-approximation algorithm by the following simple modification of Reduce-Max. We cut next the fastest growing bamboo among those with the current heights at least $x \cdot H$, for some constant $x > 1$. We call this algorithm Reduce-Fastest(x) and show the following approximation bound.

Theorem 2.1. Reduce-Fastest(2) *is a 4-approximation algorithm for discrete BGT.*

Proof. Without loss of generality, we assume that if there are two or more bamboos with the same fastest growth rate among the bamboos with the current height at least $x \cdot H$, then Reduce-Fastest chooses for trimming the bamboo with the smallest index. Thus the largest height of bamboo b_1 is at most $xH + h_1 \leq (x+1)H$.

We consider now a bamboo b_i, for some arbitrary $2 \leq i \leq n$, and assume that it reaches the height at least $C \cdot H$ for some constant $C \geq x + 1$. At any time the heights of bamboos belong to two disjoint regions: the lower region $[0, x \cdot H)$ and the upper region $[x \cdot H, \infty)$. At some point bamboo b_i must stay in the upper region for at least $\lfloor \frac{(C-x)\cdot H}{h_i} \rfloor$ consecutive rounds to reach the height $C \cdot H$.

We consider a period of $t = \lfloor \frac{(C-x)\cdot H}{h_i} \rfloor$ consecutive rounds when bamboo b_i remains in the upper region. At each of these rounds, trimming of bamboo b_i "is blocked" by trimming of another bamboo b_j for some $j < i$. The number of times when bamboo b_j can block bamboo b_i during this period is at most $t_j = 1 + \lfloor \frac{t}{f_j} \rfloor$, where $f_j = \lceil \frac{x \cdot H}{h_j} \rceil$

is the number of rounds needed by bamboo b_j to climb back to the upper region after trimming. Thus the number of rounds when bamboo b_i is blocked is at most

$$\sum_{j=1}^{i-1} t_j = \sum_{j=1}^{i-1} \left(1 + \left\lfloor \frac{\lfloor \frac{(C-x)H}{h_i} \rfloor}{\lceil \frac{xH}{h_j} \rceil} \right\rfloor\right) \leq \left\lfloor \frac{(C-x)\cdot H}{h_i} \right\rfloor \left(\frac{i-1}{\lfloor \frac{(C-x)\cdot H}{h_i} \rfloor} + \sum_{j=1}^{i-1} \frac{1}{\lceil \frac{x\cdot H}{h_j} \rceil} \right)$$

Using $h_i \leq H/i$ and $\sum_{j=1}^{i-1} h_j < H$, we obtain

$$\frac{i-1}{\lfloor \frac{(C-x)\cdot H}{h_i} \rfloor} + \sum_{j=1}^{i-1} \frac{1}{\lceil \frac{x\cdot H}{h_j} \rceil} \;<\; \frac{1}{C-x} + \frac{1}{x}.$$

Bamboo b_i is blocked in all $\lfloor \frac{(C-x)\cdot H}{h_i} \rfloor$ rounds, so

$$\sum_{j=1}^{i-1} t_j \geq \left\lfloor \frac{(C-x)\cdot H}{h_i} \right\rfloor,$$

implying that

$$\frac{1}{C-x} + \frac{1}{x} > 1.$$

The above inequality is equivalent to $C < 2 + (x-1) + 1/(x-1)$. This bound is minimized for $x = 2$, giving $C < 4$. Thus the approximation ratio of Reduce-Fastest(2) is at most 4. □ □

2.2 Offline Scheduling

In this section we focus on off-line scheduling which permits tighter approximation results. We recall first classical *Pinwheel* scheduling problem which is closely related to BGT. This is followed by the presentation of a 2-approximation algorithm for any distribution of the growth rates and a $(1+\delta)$-approximation algorithm for instances with more balanced growth rates in BGT.

2.2.1 Pinwheel

The Pinwheel problem [13] is defined as follows. Given a set $V = f_1, f_2, \ldots, f_n$ of positive integers called Pinwheel frequencies. One is asked to create an infinite sequence S of indices drawn from the set $1, 2, \ldots, n$, s.t., any sub-sequence of $f_i \in V$ consecutive elements in S includes at least one index i. The density of set V is defined as $D = \sum_{i=1}^{n} \frac{1}{f_i}$. It has been coined in [13] that Pinwheel is NP-hard assuming succinct representation of the problem. It is also known [13] that all instances of Pinwheel with the density exceeding value 1 cannot be scheduled. On the other hand any instance of Pinwheel with the density at most $\frac{3}{4}$ can be scheduled, however, finding such a schedule may require a substantial time [12].

In order to determine the relationship between BGT and Pinwheel problems we show first how to relate the daily growth rates in BGT with the frequencies in Pinwheel. We define the set of frequencies $f_i = H/h_i$, for $i = 1, 2, \ldots, n$, which form a *pseudo-instance* of Pinwheel with frequencies as real numbers (rather than integers) and with the density

$$D = \sum_{i=1}^{n} \frac{1}{f_i} = \sum_{i=1}^{n} \frac{h_i}{H} = 1.$$

Note that one can replace H by $H' = (1+\delta)H$, for any $\delta > 0$ to reduce the density of the respective pseudo-instance to

$$D' = \sum_{i=1}^{n} \frac{1}{f'_i} = \sum_{i=1}^{n} \frac{h_i}{(1+\delta)H} = \frac{1}{(1+\delta)} \sum_{i=1}^{n} \frac{1}{f_i} = \frac{1}{(1+\delta)} D.$$

In other words, by manipulating δ one can obtain another pseudo-instance $I'(\delta)$ of Pinwheel with the density $\frac{1}{(1+\delta)}$ lower than one. For example, by adopting $\delta = \frac{1}{3}$ one can obtain a pseudo-instance $I'(\frac{1}{3})$ of Pinwheel with the density $\frac{3}{4}$.
Furthermore, having a pseudo-instance $I'(\delta)$ with sufficiently low density $\frac{1}{(1+\delta)}$, for $\delta > 0$, enables replacement of non-integral frequencies by their floors to create a proper instance $I(\delta)$ of Pinwheel with the density below one.

Lemma 2.1. *A solution (if feasible) to the proper instance $I(\delta)$ of Pinwheel results in a $(1+\delta)$-approximation schedule for the original BGT problem.*

Proof. In $I(\delta)$ the frequence $f_i \leq \frac{H(1+\delta)}{h_i}$ is an upper bound on the number of rounds between two consecutive visits to b_i in BGT. And since the height of b_i is limited to $h_i \cdot f_i$ we get the upper bound $H(1+\delta)$ on the height of each b_i. □ □

A 2-approximation algorithm. According to Lemma 2.1 the main challenge in BGT, i.e., keeping all bamboos as low as possible can be reduced to finding the smallest value of δ for which the relevant proper instance of Pinwheel problem can be scheduled. The main idea behind our solution refers to the result from [13] indicating that any instance of Pinwheel with frequencies being powers of 2 and the density at most 1 can be scheduled efficiently. By adopting $H' = 2H$ one can first translate any instance of BGT to a pseudo-instance of Pinwheel with the density $\frac{1}{2}$, and later by reducing each frequency to the nearest power of 2 produce a proper instance of Pinwheel with the density at most 1.

Corollary 2.1. *The algorithm described above provides a 2-approximation for the BGT problem.*

A $(1+\delta)$-approximation algorithm for more balanced growth rates.
In search for more tight approximation one cannot reduce frequencies to just the closest power of 2. Instead, to obtain greater granularity we start with reduction of frequencies (in the respective pseudo-instance of Pinwheel) to the closest values of the form $2^k(1+\frac{j}{C})$, where $C = 2^a$, for some integer constant $a \geq 0$, and $j \in [0, C)$. We make the following two observations.

Observation 1. Any two frequencies of the form $2^k(1+\frac{j}{C})$ can be combined via their equidistant superposition into a shorter frequency $2^{k-1}(1+\frac{j}{C})$. For example, for $k = 4, C = 4$ and $j = 3$ we obtain two frequencies f_1, f_2 of size $2^k(1+\frac{j}{C}) = 2^4(1+\frac{3}{4}) = 28$ which can be combined into a shorter frequency $2^{k-1}(1+\frac{j}{C}) = 2^3(1+\frac{3}{4}) = 14$ by alternating f_1 and f_2 in a round robin fashion.

Observation 2. One can combine $m_j = C + j$ frequencies $2^k(1+\frac{j}{C})$ into one frequency $2^k/C$ which is also a power of 2, since $2^k(1+\frac{j}{C})/m_j = 2^k/C$.

We say that an instance of BGT is *α-balanced*, if $h_1 \leq \alpha \cdot H$, for some constant

$\alpha < 1$.

The Main Algorithm. Given an α-balanced instance of BGT with growth rates $h_1, h_2, \ldots, h_n$.

1. Adopt $H' = (1+\delta)H$, and form the respective pseudo-instance of Pinwheel with the frequencies $f_1, f_2, \ldots, f_n > 2^{min}$, for the largest possible integer min, and the density $\frac{1}{1+\delta}$.

2. Reduce each frequency f_i to the closest value of the form $2^k(1+\frac{j}{C})$, for some $k \geq min$ and $j \in [0, C)$.
   [*This increases the density by a factor of* $1+\frac{1}{C}$ *to the value* $(1+\frac{1}{C})/(1+\delta)$.]

3. Use Observation 1 for as long as possible to combine pairs of the same frequencies pushing them down towards the range $[2^{min}, 2^{min+1})$.
   [*On the conclusion of this step there is at most one frequency* $2^k(1+\frac{j}{C})$, *for* $k > min$ *and* $j \in [0, C)$.]

4. Apply the transformation from Observation 2 in the range $[2^{min}, 2^{min+1})$ until there is at most $C + j - 1$ frequencies $2^k(1+\frac{j}{C})$ left, for any $j \in (0, C)$.
   [*After this step, there are at most* $C + j - 1$ *frequencies in each group* j *in the range* $[2^{min}, 2^{min+1})$.]

5. In each range reduce all remaining frequencies (different to powers of two) group by group starting from the top group $j = C - 1$ and apply the transformation from Observation 2 whenever possible.
   [*We gain an extra density* ΔD. *We must ensure that* $\frac{1+\frac{1}{C}}{1+\delta} + \Delta D \leq 1$. *This can be done by the appropriate selection of parameters* C *and* δ, *see below.*]

The following theorem about the approximation of the above algorithm is proven in the full version of the paper.

Theorem 2.2. *For any* $\delta > 0$, *the Main Algorithm produces* $(1+\delta)$*-approximation BGT schedules for* α*-balance instances, if* $\alpha \leq \frac{\delta^2(1+\delta)}{(2+\delta)^2}$.

3 Continuous BGT

We consider now the continuous variant of the BGT problem. Since this variant models scenarios when bamboos are spread over some geographical area, we will now refer not only to bamboos $b_1, b_2, \ldots, b_n$ but also to the points $v_1, v_2, \ldots, v_n$ (in the implicit underlying space) where these bamboos are located. We will denote by V the set of these points.

Recall that input I for the continuous BGT problem consists of the rates of growth of bamboos $(h_i : 1 \leq i \leq n)$ and the travel times between bamboos $(t_{i,j} : 1 \leq i, j \leq n)$. We assume that $h_1 \geq h_2 \geq \ldots \geq h_n$, as before, and normalize these rates, for convenience, so that $h_1 + h_2 + \ldots + h_n = 1$ (this is done without loss of generality, since the exact unit of the heights of bamboos is irrelevant). We assume that the travel distances are symmetric and form a metric on V. (In the scenarios which we model, if $t_{i,j}$ was greater than $t_{i,k} + t_{k,j}$, then the robot would travel between points v_i and v_j via the point v_k.)

Algorithm 1: An $O(h_{\max}/h_{\min})$-approximation algorithm for continuous BGT.

1. Calculate a minimum spanning tree T of the point set V.
2. Repeatedly perform an Euler-tour traversal of T.

For any $V' \subseteq V$, the minimum growth rate among all points in V' is denoted by $h_{\min}(V')$, and the maximum growth rate among all points in V' is denoted by $h_{\max}(V')$. Let $h_{\min} = h_{\min}(V) = h_n$, and $h_{\max} = h_{\max}(V) = h_1$.

The diameter of the set V is denoted by $D = D(V) = \max\{t_{i,j} : 1 \leq i, j \leq n\}$. For any $V' \subseteq V$, $MST(V')$ denotes the minimum weight of a Steiner tree on V'. Recall that for an algorithm $\mathcal{A}$ and input I, $MH(\mathcal{A}(I))$ denotes the maximum height that any bamboo ever reaches, if trimming is done according to the schedule computed by $\mathcal{A}$, and $OPT(I)$ is the optimal (minimal) maximum height of a bamboo over all schedules.

3.1 Lower bounds

We first show some simple lower bounds on the maximum height of a bamboo. For notational brevity, we omit the explicit reference to the input I. For example, the inequality $MH(\mathcal{A}) \geq Dh_{\max}$ in the lemma below is to be understood as $MH(\mathcal{A}(I)) \geq D(V(I)) \cdot h_{\max}(V(I))$, for each input instance I.

Lemma 3.1. *$MH(\mathcal{A}) \geq Dh_{\max}$, for any algorithm $\mathcal{A}$.*

Proof. The robot must visit another point x in V at distance at least $D/2$ from v_1. When the robot comes back to v_1 after visiting x (possibly via a number of points in V), the bamboo at v_1 has grown at least to the height of Dh_1. □ □

Lemma 3.2. *$MH(\mathcal{A}) = \Omega(h_{\min}(V') \cdot MST(V'))$, for any algorithm $\mathcal{A}$ and $V' \subseteq V$.*

Proof. Let v be the point in V' visited last: all points in $V' \setminus \{v\}$ have been visited at least once before the first visit to v. The distance traveled until the first visit to v is at least $MST(V')$, so the bamboo at v has grown to the height at least $h_v \cdot MST(V')$. □□

3.2 Approximation algorithms

We describe our Algorithms 1, 2 and 3 for the continuous BGT problem in pseudocode and give their approximation ratio in the theorems below.

Theorem 3.1. *Algorithm 1 is an $O(h_{\max}/h_{\min})$-approximation algorithm for the continuous BGT problem.*

Proof. Let $\mathcal{A}_1$ denote Algorithm 1. Every point $v_i \in V$ is visited by $\mathcal{A}_1$ at least every $2 \cdot MST(V)$ time units. Hence,

$$MH(\mathcal{A}_1) = O(h_{\max}(V) \cdot MST(V)). \tag{1}$$

According to Lemma 3.2,

$$OPT = \Omega(h_{\min}(V) \cdot MST(V)). \tag{2}$$

Algorithm 2: An $O(\log(h_{\max}/h_{\min}))$-approximation algorithm for continuous BGT.

1. Let $s = \lceil \log_2(h_{\max}/h_{\min}) \rceil$.
2. For $i \in \{1, 2 \ldots, s\}$, let $V_i = \{v_j \in V \mid 2^{i-1} \cdot h_{\min} \leq h_j < 2^i \cdot h_{\min}\}$, let T_i be an $O(1)$-approximation of the minimum Steiner tree on V_i, and let C_i be an Euler-tour traversal of T_i.
3. For $i \in \{2, \ldots, s\}$, define an arbitrary point on C_i as the last visited point on C_i.
4. Start at an arbitrary point on C_1.
5. **repeat forever**
6. **for** $i = 1$ **to** $s - 1$ **do**
7. Walk distance D on C_i in clockwise direction.
8. Walk to the last visited point on C_{i+1}.
9. **for** $i = s$ **to** 2 **do**
10. Walk distance D on C_i in clockwise direction.
11. Walk to the last visited point on C_{i-1}.

Combining the two bounds (1) and (2), it follows that Algorithm 1 is an $O(h_{\max}/h_{\min})$-approximation algorithm for BGT. □
□

Theorem 3.2. *Algorithm 2 is an $O(\log(h_{\max}/h_{\min}))$-approximation algorithm for the continuous BGT problem.*

Proof. Consider any point $v \in V_i$, for any $i \in \{1, 2, \ldots, s\}$. The distance traveled between two consecutive visits to v is at most

$$O\left(D \cdot \log\left(\frac{h_{\max}}{h_{\min}}\right) \cdot \left\lceil \frac{MST(V_i)}{D} \right\rceil\right) = O\left(\log\left(\frac{h_{\max}}{h_{\min}}\right) \cdot \max\{D, MST(V_i)\}\right).$$

Hence, the height of the bamboo at v is never larger than

$$O\left(h_{\max}(V_i) \cdot \log\left(\frac{h_{\max}}{h_{\min}}\right) \cdot \max\{D, MST(V_i)\}\right). \tag{3}$$

On the other hand, using Lemmas 3.1 and 3.2, we obtain

$$OPT = \Omega(h_{\min}(V_i) \cdot \max\{D, MST(V_i)\}). \tag{4}$$

Combining the two bounds (3) and (4), and observing that $h_{\max}(V_i) \leq 2 \cdot h_{\min}(V_i)$, we see that Algorithm 2 is an $O(\log(h_{\max}/h_{\min}))$-approximation algorithm for BGT. □ □

Algorithm 3: An $O(\log n)$-approximation algorithm for continuous BGT.

1. Let $s = \lceil 2 \cdot \log_2 n \rceil$.
2. Let $V_0 = \{v_i \in V \mid h_i \leq n^{-2}\}$.
   For $i \in \{1, 2, \ldots, s\}$, let $V_i = \{v_j \in V \mid 2^{i-1} \cdot n^{-2} < h_j \leq 2^i \cdot n^{-2}\}$.
   For $i \in \{1, 2, \ldots, s\}$, let T_i be an $O(1)$-approximation of the minimum Steiner tree on V_i, and let C_i be an Euler-tour traversal of T_i.
3. For $i \in \{2, 3, \ldots, s\}$, define an arbitrary point on C_i as the last visited point on C_i. Let $V_0 = \{v'_0, v'_1, \ldots, v'_{\ell-1}\}$.
4. Start at an arbitrary point on C_1.
5. $j = 0$.
6. **repeat forever**
7. **for** $i = 1$ **to** $s - 1$ **do**
8. Walk distance D on C_i in clockwise direction.
9. Walk to the last visited point on C_{i+1}.
10. **for** $i = s$ **to** 2 **do**
11. Walk distance D on C_i in clockwise direction.
12. Walk to the last visited point on C_{i-1}.
13. Walk to $v'_{j \bmod \ell}$ and back.
14. $j = j + 1$.

Theorem 3.3. *Algorithm 3 is an $O(\log n)$-approximation algorithm for the continuous BGT problem.*

Proof. Consider any point $v \in V_i$, for any $i \in \{1, 2, \ldots, s\}$. Then, the distance traveled between two consecutive visits of v is at most

$$O(D \cdot \log n \cdot \left\lceil \frac{MST(V_i)}{D} \right\rceil) = O(\log n \cdot \max\{D, MST(V_i)\}).$$

Hence, the height of the bamboo at v is never larger than

$$O(h_{\max}(V_i) \cdot \log n \cdot \max\{D, MST(V_i)\}). \tag{5}$$

On the other hand, using Lemmas 3.1 and 3.2, we obtain

$$OPT = \Omega(h_{\min}(V_i) \cdot \max\{D, MST(V_i)\}). \tag{6}$$

Since $h_{\max}(V_i) \leq 2h_{\min}(V_i)$, then the height of the bamboo at v is always $O(OPT \cdot \log n)$.

Consider a point $v \in V_0$. Then, the distance traveled between two consecutive visits of v is at most

$$O(|V_0| \cdot D \cdot \log n) = O(n \cdot D \cdot \log n).$$

Hence, the height of the bamboo at v is never larger than

$$O(n \cdot h_{\max}(V_0) \cdot D \cdot \log n) = O(n \cdot n^{-2} \cdot D \cdot \log n) = O(h_{\max} \cdot D \cdot \log n). \tag{7}$$

On the other hand, using Lemma 3.1, we obtain

$$OPT = \Omega(h_{\max} \cdot D), \tag{8}$$

so the height of the bamboo at a point in V_0 is also always $O(OPT \cdot \log n)$. Thus Algorithm 3 is an $O(\log n)$-approximation algorithm for BGT. □ □

4 Open Problems

There are several interesting open questions about approximation algorithms for the BGT problems, including better understanding of the approximation ratio of Reduce-Max for discrete BGT. For continuous BGT, we do not know whether our Algorithm 3 or any other algorithm achieves an approximation ratio $o(\log n)$. There are also questions about efficient implementation of BGT algorithms. For example, how can we select the highest bamboo in Reduce-Max faster than in linear time per round, if the growth rates are known to the gardener?

References

[1] S Alshamrani, D.R. Kowalski, L. Gąsieniec, How Reduce Max Algorithm Behaves with Symptoms Appearance on Virtual Machines in Clouds, In Proc. *IEEE International Conference* CIT/IUCC/DASC/PICOM, 2015, pp 1703-1710.

[2] S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D. A. Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 15(6):600–625, June 1996.

[3] S. K. Baruah and S.-S. Lin. Pfair scheduling of generalized pinwheel task systems. *IEEE Transactions on Computers*, 47(7):812–816, July 1998.

[4] M. A. Bender, S. P. Fekete, A. Kröller, J.S.B. Mitchell, V. Liberatore, V. Polishchuk, J. Suomela. The Minimum Backlog Problem. *Theoretical Computer Science*, 605:51–61, November 2015.

[5] M. H. L. Bodlaender, C. A. J. Hurkens, V. J. J. Kusters, F. Staals, G. J. Woeginger, and H. Zantema Cinderella versus the Wicked Stepmother. In Proc. *TCS 2012*, LNCS v. 6942, pages 57–71. Springer, 2012.

[6] M. Y. Chan and F. Y. L. Chin. General schedulers for the pinwheel problem based on double-integer reduction. *IEEE Transactions on Computers*, 41(6):755–768, June 1992.

[7] M. Y. Chan and F. Chin. Schedulers for larger classes of pinwheel instances. *Algorithmica*, 9(5):425–462.

[8] M. Chrobak, J. Csirik, C. Imreh, J. Noga, J. Sgall, G.J. Woeginger. The Buffer Minimization Problem for Multiprocessor Scheduling with Conflicts. In Proc. *ICALP 2001*, LNCS v. 2076, pages 862–874. Springer 2001.

[9] A. Collins, J. Czyzowicz, L. Gąsieniec, A. Kosowski, E. Kranakis, D. Krizanc, R. Martin, and O. Morales Ponce. Optimal Patrolling of Fragmented Boundaries. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '13, pages 241–250, New York, USA, 2013.

[10] J. Czyzowicz, L. Gąsieniec, A. Kosowski, and E. Kranakis. Boundary Patrolling by Mobile Agents with Distinct Maximal Speeds. In Proc. *ESA 2011*, number 6942 in Lecture Notes in Computer Science, pages 701–712. Springer, September 2011.

[11] J. Czyzowicz, L. Gąsieniec, A. Kosowski, E. Kranakis, D. Krizanc, and N. Taleb. When Patrolmen Become Corrupted: Monitoring a Graph using Faulty Mobile Robots In Proc. ISAAC 2015, 26th International Symposium on Algorithms and Computation, pp. 343-354.

[12] P.C. Fishburn and J,C, Lagarias, Pinwheel Scheduling: Achievable Densities. *Algorithmica*, 34(1):14–38.

[13] R. Holte, A. Mok, L. Rosier, I. Tulchinsky, and D. Varvel. The pinwheel: a real-time scheduling problem. In *II: Software Track, Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences, 1989. Vol*, volume 2, pages 693–702 vol.2, January 1989.

[14] R. Holte, L. Rosier, I. Tulchinsky, and D. Varvel. Pinwheel scheduling with two distinct numbers. *Theoretical Computer Science*, 100(1):105–135, June 1992.

[15] A. Kawamura and Y. Kobayashi, Fence patrolling by mobile agents with distinct speeds, Distributed Computing 28(2): 147-154, 2015.

[16] S.-S. Lin and K.-J. Lin. A Pinwheel Scheduler for Three Distinct Numbers with a Tight Schedulability Bound. *Algorithmica*, 19(4):411–426.

[17] S. Ntafos. On gallery watchmen in grids. *Information Processing Letters*, 23(2):99–102, 1986.

[18] T. H. Romer and L. E. Rosier. An algorithm reminiscent of euclidean-gcd for computing a function related to pinwheel scheduling. *Algorithmica*, 17(1):1–10, 1997.

[19] P. Serafini and W. Ukovich. A Mathematical Model for Periodic Scheduling Problems. *SIAM Journal on Discrete Mathematics*, 2(4):550–581, November 1989.

[20] J. Urrutia. Art gallery and illumination problems. *Handbook of computational geometry*, 1(1):973–1027, 2000.