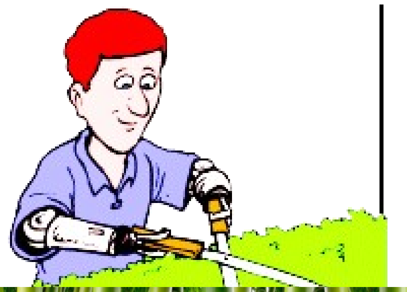
Bamboo Garden Trimming Problem

Ralf Klasing, CNRS – LaBRI – Univ. of Bordeaux

Joint work with Leszek A Gąsieniec, Christos Levcopoulos,

Andrzej Lingas, Jie Min, and Tomasz Radzik

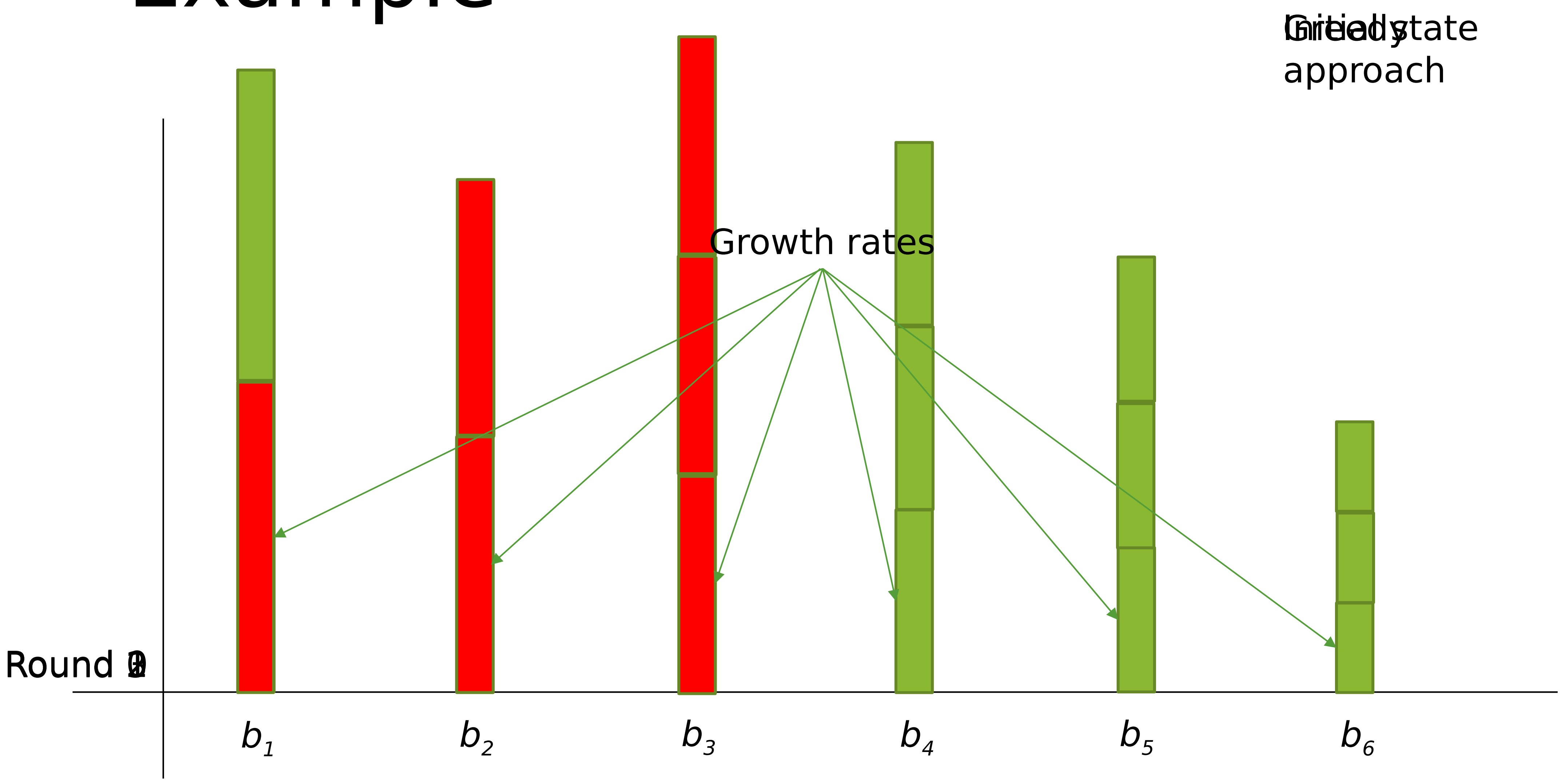
Bamboo Garden Trimming



- Given a set B of n bamboos $b_1, b_2,\ldots,b_n$ with the respective (daily) growth rates $h_1 \geq h_2 \geq \cdots \geq h_n$, where the initial heights of all bamboos are set to *zero*.
- **Discrete BGT**
  - During each round/day every bamboo b_i grows an extra height h_i and on the conclusion of the round the height of exactly one bamboo is reduced to *zero*.
- **Continuous BGT**
  - It requires time $t_{ij}>0$ for the gardener to move from b_i to b_j and the travel distances are symmetric.
  - When attended the bamboos are cut instantly.
  - **Note:** - Discrete BGT is a special case of Continuous BGT when $t_{ij} = 1$ for all i,j

    - Metric TSP is a special case of Continuous BGT when $h_i = h_j$ for all i,j
- The **main task** in **BGT Problem** is to find a perpetual schedule of cuts with the goal of keeping the height of the bamboo garden as low as possible.

Example


Initial state
Greedy approach
Growth rates
Round 0
b1
b2
b3
b4
b5
b6

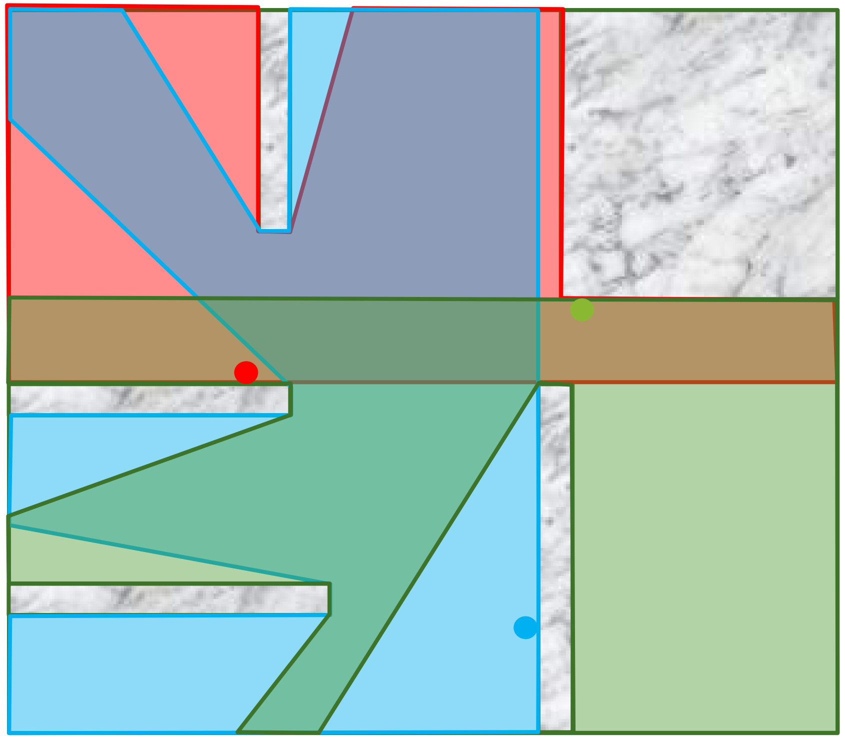
The main results

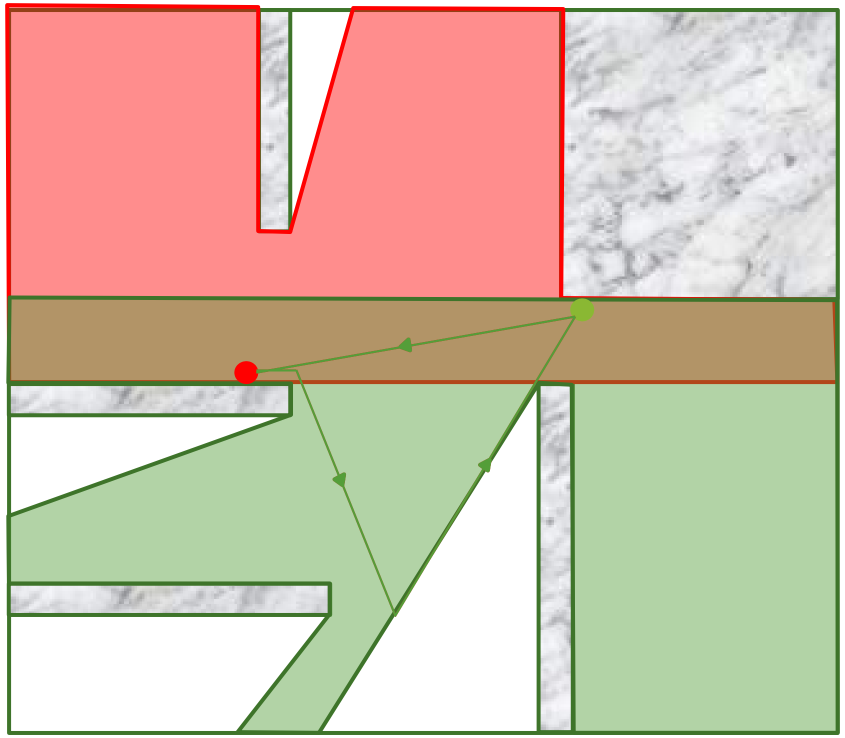
Algorithms	Approximation ratio	Time complexity in #steps
Discrete BGT		
Reduce-Max (Greedy approach)	$O(\log_2 n)$	*O(n log n)* + each round in *O(log² n)*
Reduce-Fastest	4	*O(n log n)* + each round in *O(log n)*
Reduce-to-Pinwheel	2 1+δ, balanced instances	*O(n)* + each round in *O(1)* *O(n log n)* + each round in *O(log n)*
Continuous BGT		
Algorithm 1	$O(\frac{h_1}{h_n})$	Generation of MST *O(n²)* + each round in *O(1)*
Algorithm 2	$O(\log_2 \frac{h_1}{h_n})$	Partition *O(n)* + Generation of MST *O(n²)*
Algorithm 3	$O(\log_2 n)$	Partition *O(n)* + Generation of MST *O(n²)*

Note: We do not know whether BGT is tractable, similarly to closely related Pinwheel scheduling studied for years

Research context

- The Art Gallery Problem



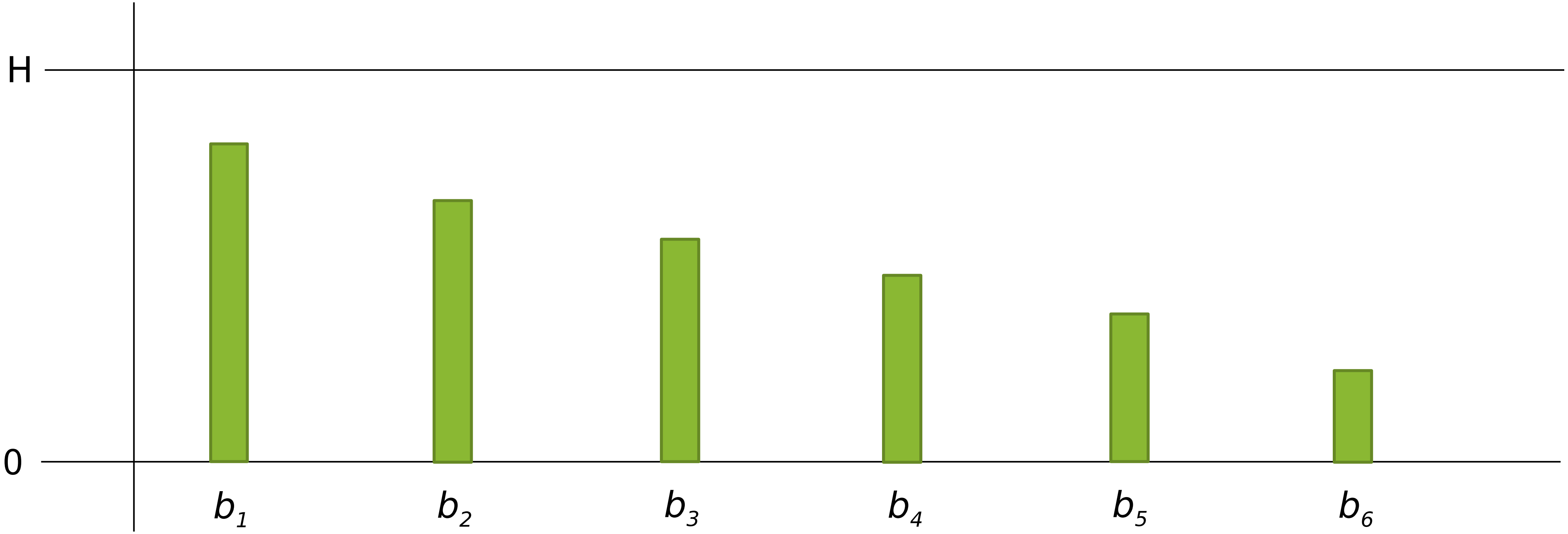


- K-watchman Problem
- Boundary (Fence) Patrolling Problem
- Cloud computing
  - Symptom discovery, origin of our problem
- Pinwheel problem (to be discussed later)

Discrete BGT: Important lower bound H

- Max height cannot be kept below $H = h_1 + h_2 + \cdots + h_n$, i.e., a single **round contribution**

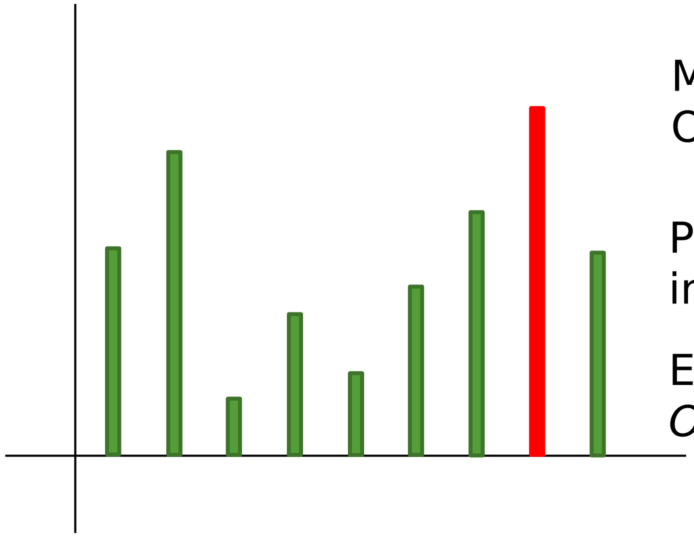
[The argument is based on the total height which keeps increasing when the cuts are < H.]



Discrete BGT

- **Online**: memoryless, flexible (fault-tolerant, self-stabilising), harder to analyse

- *Reduce-Max* (Greedy 1)

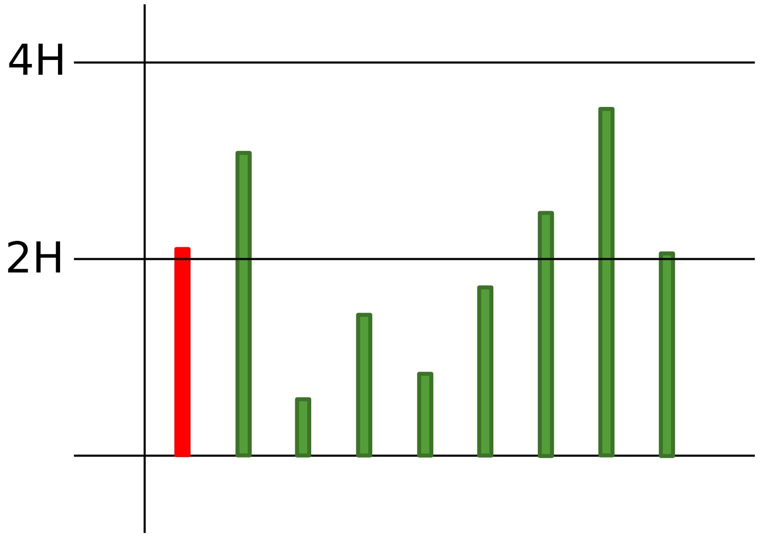


Max height $O(H \log n)$

Preprocessing in $O(n \log n)$

Each round in $O(\log^2 n)$ steps

- *Reduce-Fastest* (Greedy 2)



Max height $\leq 4H$

Preprocessing in $O(n \log n)$

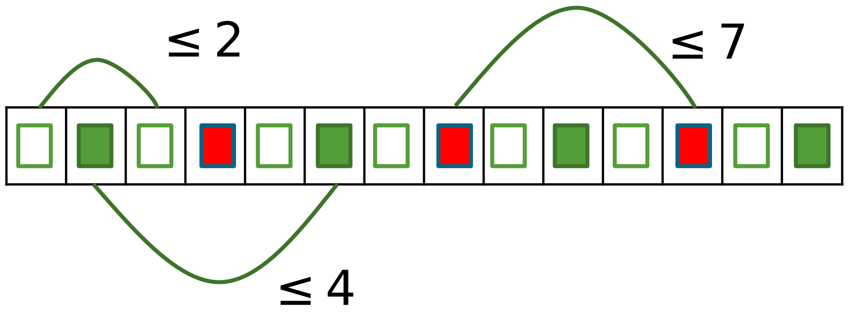
Each round in $O(\log n)$ steps

- Approximation ratio μ defined as *Maxheight / H*
- Until recently $\mu = O(\log n)$ for *Reduce-Max* and $\mu = O(1)$ for *Reduce-Fastest*

Discrete BGT

- **Offline**: less flexible (vulnerable to changes), better (more accurate) approximation μ
- Strong relation to the **Pinwheel Problem** (classical scheduling problem)

	frequency	**density**
	f	d
	2	1/2
	4	1/4
	7	1/7



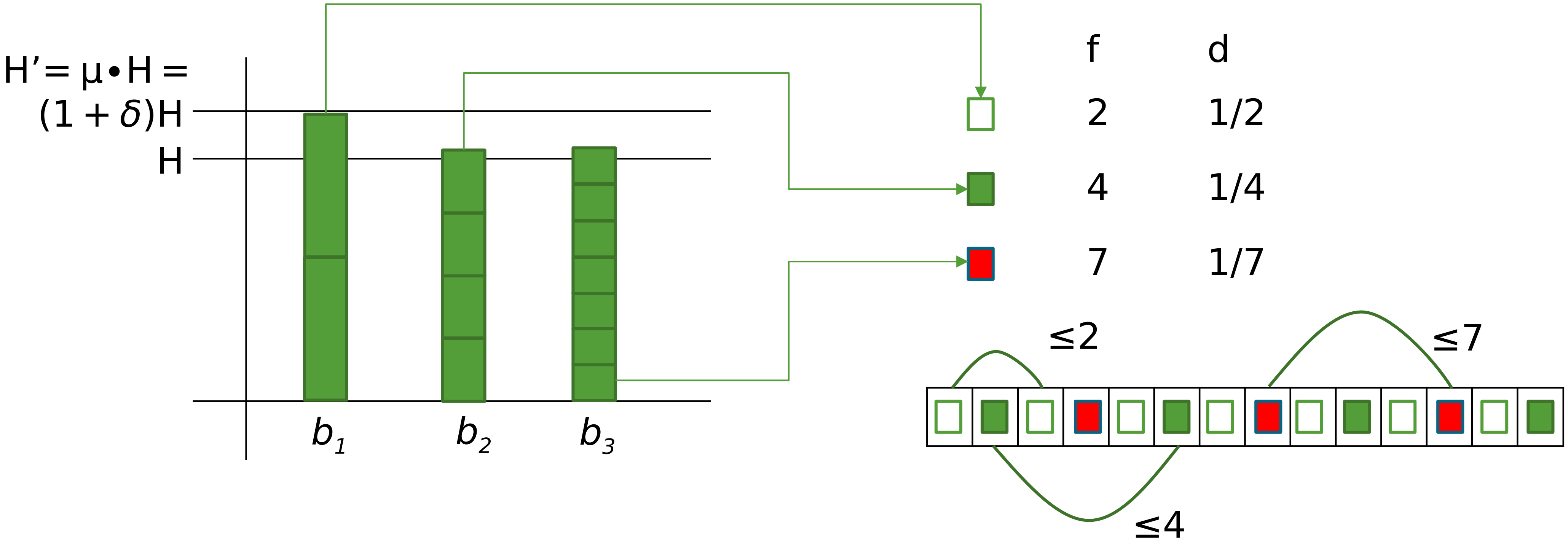
Density of the instance
D=1/2+1/4+1/7=**25/28**

Basic facts:
D>1, not schedulable
D≤1/2, easy to schedule
D≤3/4, sometimes hard to schedule
D=1, easy to schedule if all fs are powers of 2
D>5/6, not schedulable for, e.g., f=2,3, large int. N

Discrete BGT

- Reduction to Pinwheel with density ≤ 1

Comment:
By adopting δ=1 one can allow all frequencies f to be powers of 2.
I.e., **μ=2 approx. is always feasible**



Discrete BGT (μ= 1+δ, balanced instances)

- Idea:

  - derive an appropriate powers-of two instance by gradual transformations of the Pinwheel instances
  - start with greater granularity of frequencies than powers of two

- Observation1:

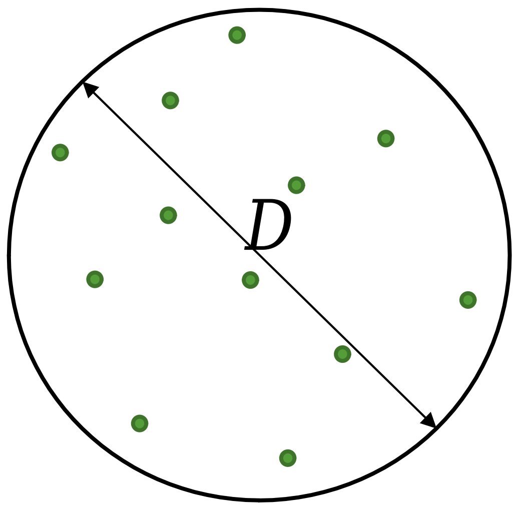
  - Given instance *V* of Pinwheel with two equal frequencies $f_i = f_j = 2f$ (f integer).
  - If the instance *V'* obtained from *V* by replacing these two frequencies with one frequency f is feasible, then so is instance *V*.

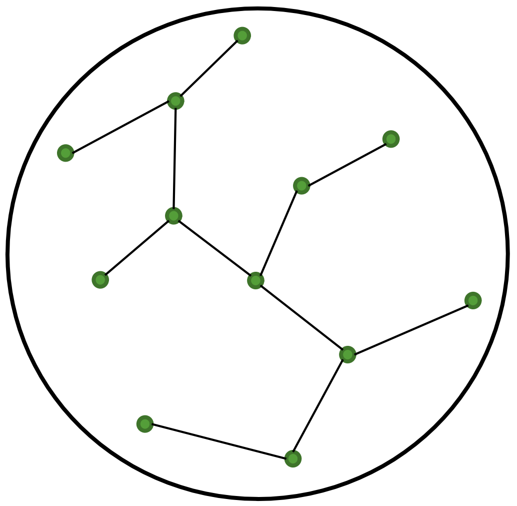
- Observation 2:

  - Given instance *V* with m equal frequencies $f_{i,1} = f_{i,2} = \ldots = f_{i,m} = mf$ (f integer).
  - If the instance *V'* obtained from *V* by replacing these m frequencies with one frequency f is feasible, then so is instance *V*.

Continuous BGT

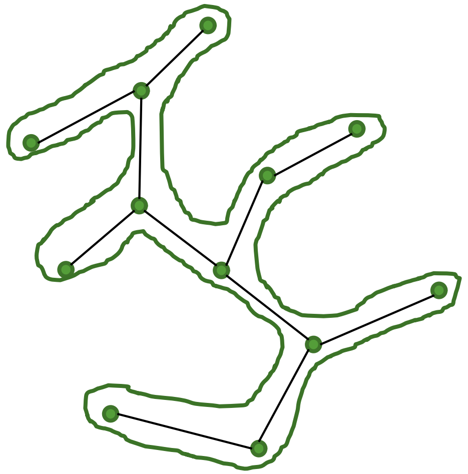
- Differences to discrete BGT:
  - Bamboos are located in a (geo)metric space of diameter *D*.
  - For any pair of bamboos b_i and b_j , the gardener needs time $t_{ij} > 0$ to travel from b_i to b_j and the travel distances are symmetric.
  - Each bamboo is cut instantly.
- Two natural lower bounds:
  - $D\, h_{max}(V)$
  - $h_{min}(V)\, MST(V)$





Continuous BGT

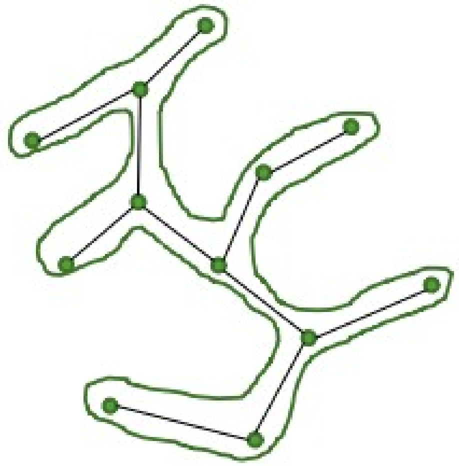
- Algorithm 1
  - Calculate a minimum spanning tree T of the bamboos.
  - Repeatedly perform an Euler-tour traversal of T.



- Upper bound: O($h_{max}(V)\, MST(V)$)

Continuous BGT

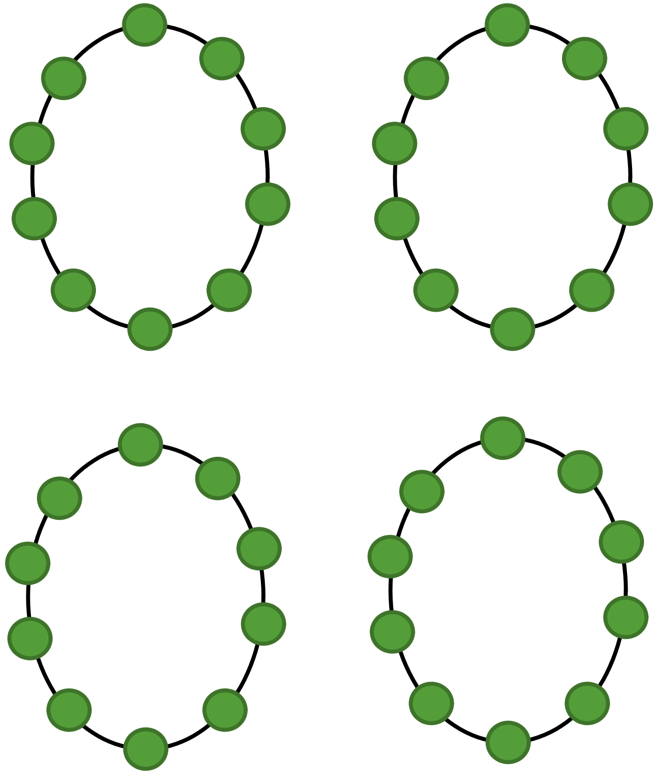
- Algorithm 1
  - Calculate a minimum spanning tree T of the bamboos.
  - Repeatedly perform an Euler-tour traversal of T.



- Upper bound: O($h_{max}(V)\ MST(V)$)
- Approximation ratio: O(h_{max}(V) / h_{min}(V))

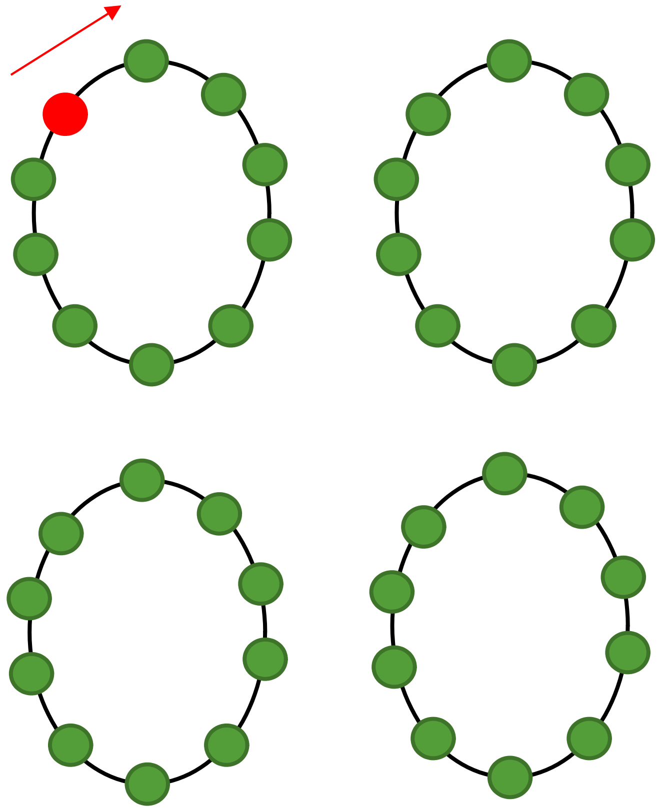
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \leq h_i < 2h_{min}$
    - V_2: $2h_{min} \leq h_i < 4h_{min}$
    - …
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



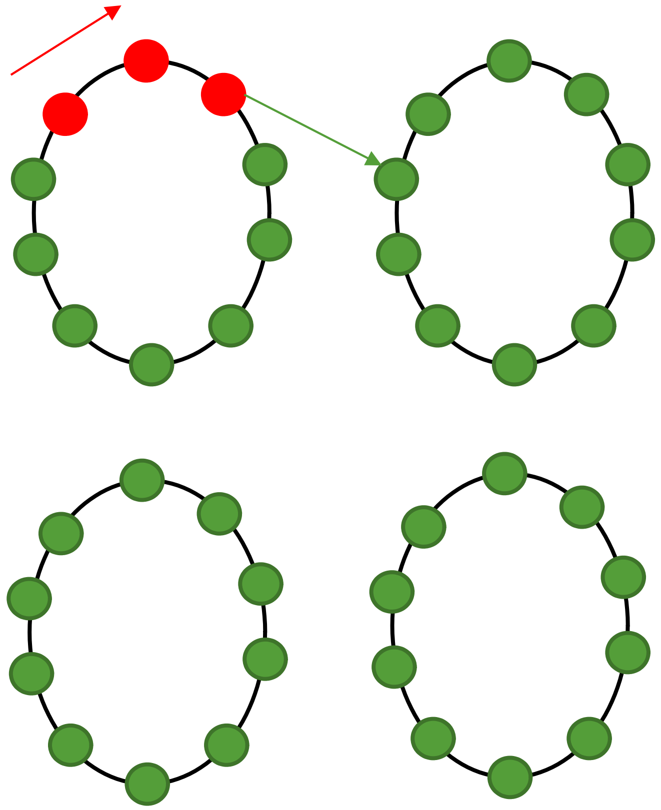
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \leq h_i < 2h_{min}$
    - V_2: $2h_{min} \leq h_i < 4h_{min}$
    - ...
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



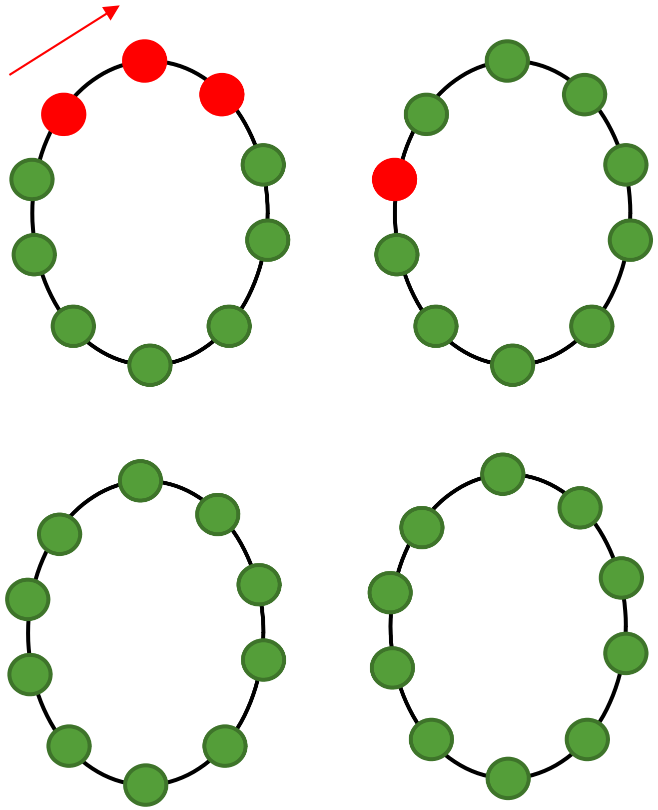
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V1: $h_{min} \leq h_i < 2h_{min}$
    - V2: $2h_{min} \leq h_i < 4h_{min}$
    - ...
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



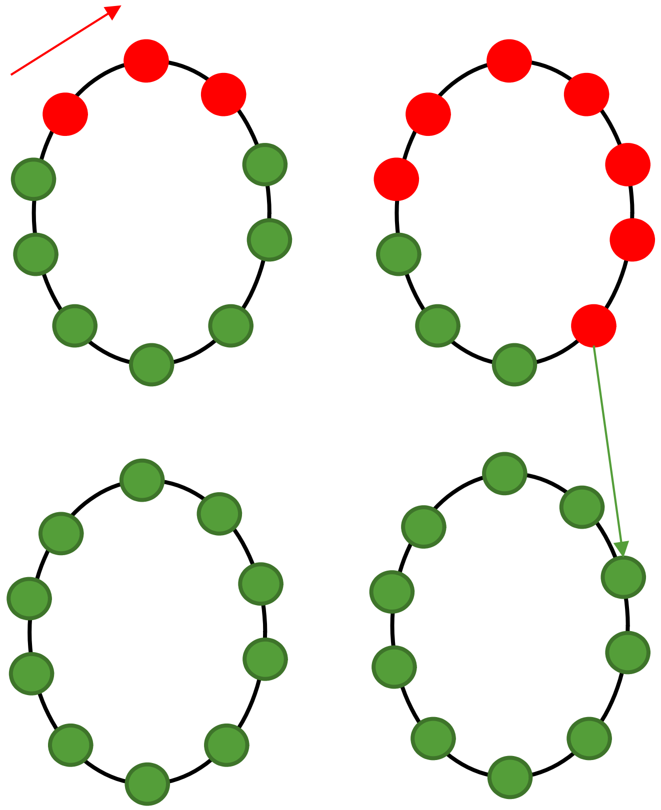
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \le h_i < 2h_{min}$
    - V_2: $2h_{min} \le h_i < 4h_{min}$
    - …
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



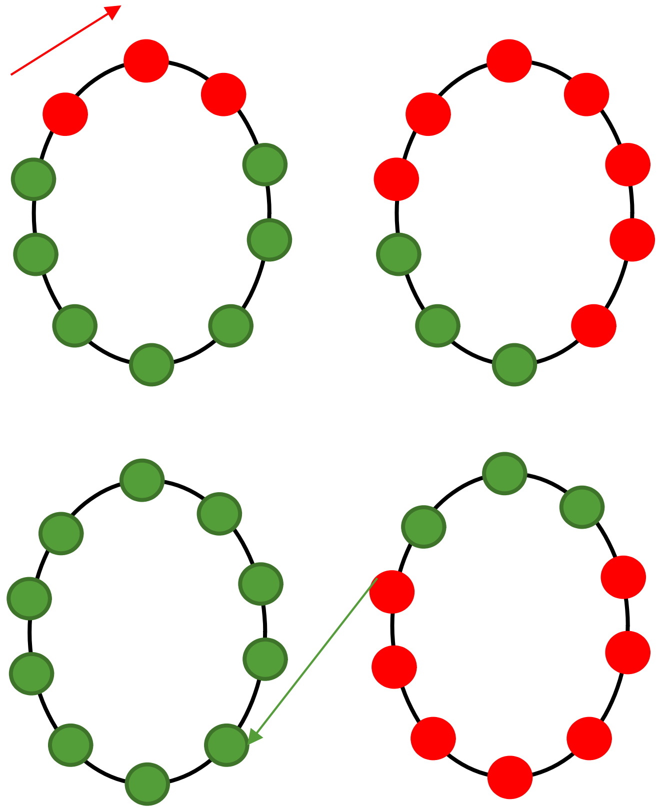
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \leq h_i < 2h_{min}$
    - V_2: $2h_{min} \leq h_i < 4h_{min}$
    - …
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



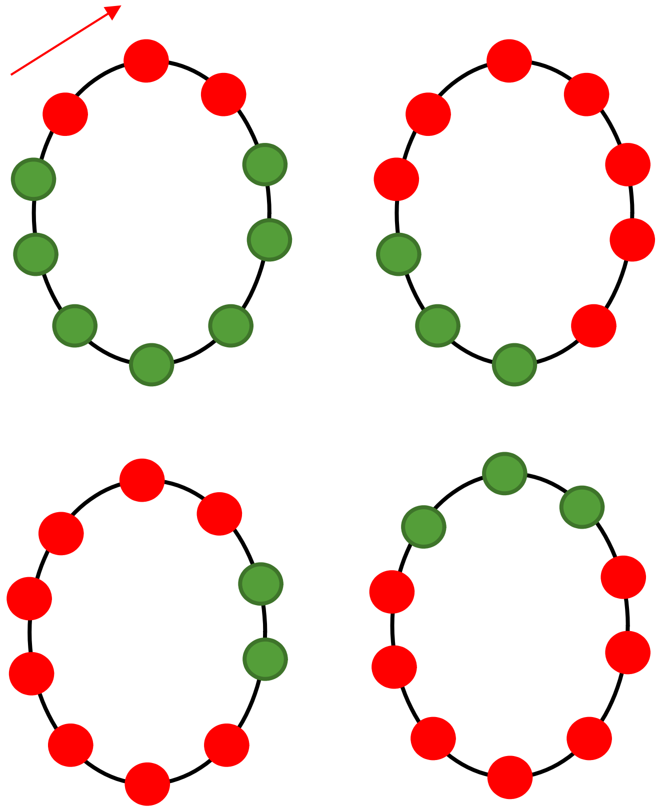
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \leq h_i < 2h_{min}$
    - V_2: $2h_{min} \leq h_i < 4h_{min}$
    - …
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



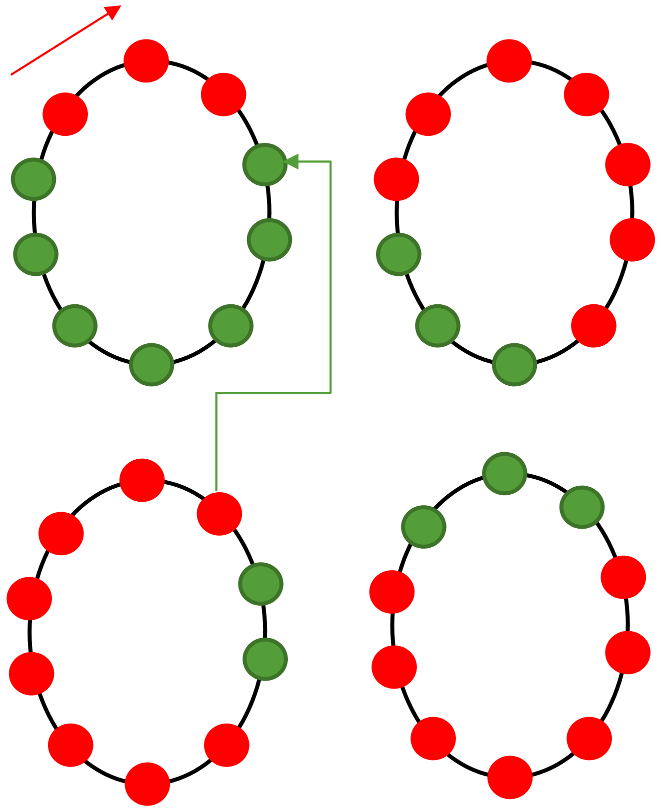
Continuous BGT

- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \leq h_i < 2h_{min}$
    - V_2: $2h_{min} \leq h_i < 4h_{min}$
    - …
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



Continuous BGT

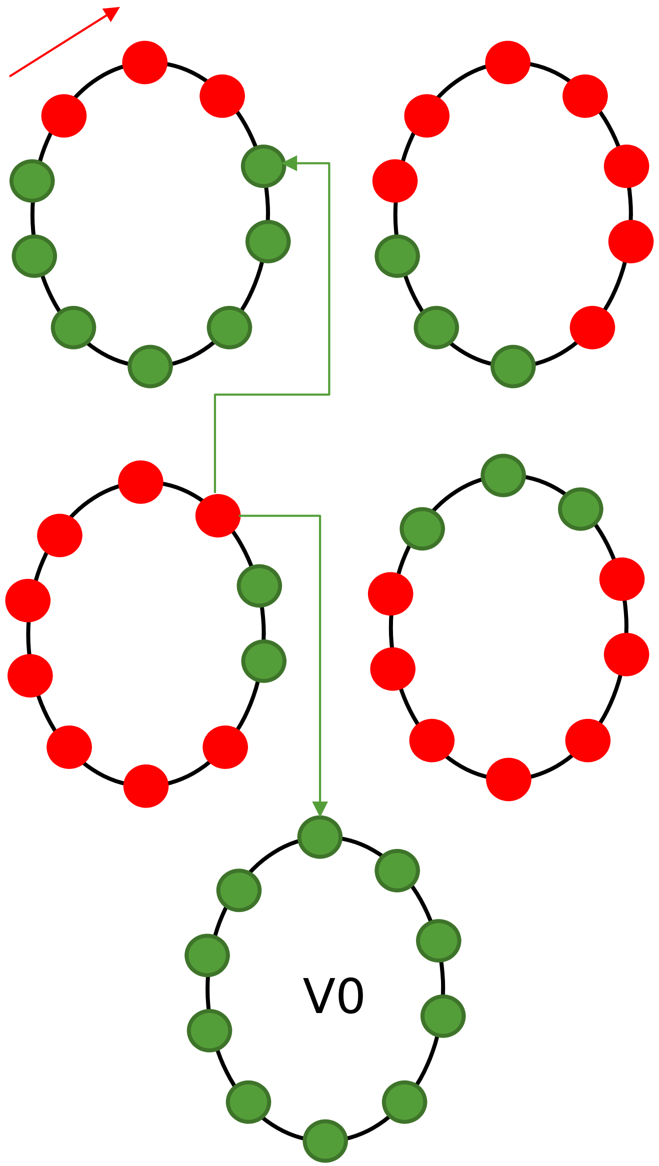
- Algorithm 2
  - Partition bamboos into sets according to their growth rates.
    - V_1: $h_{min} \le h_i < 2h_{min}$
    - V_2: $2h_{min} \le h_i < 4h_{min}$
    - …
    - At most $\left\lceil \log_2 \frac{h_{max}}{h_{min}} \right\rceil$ sets
  - Calculate a minimum spanning tree T of the bamboo sets.



- Upper bound: $O(\log_2 \frac{h_{max}}{h_{min}} h_{max}(V_i) \max\{D, MST(V_i)\})$
- Approximation ratio: $O(\log_2 \frac{h_{max}}{h_{min}})$

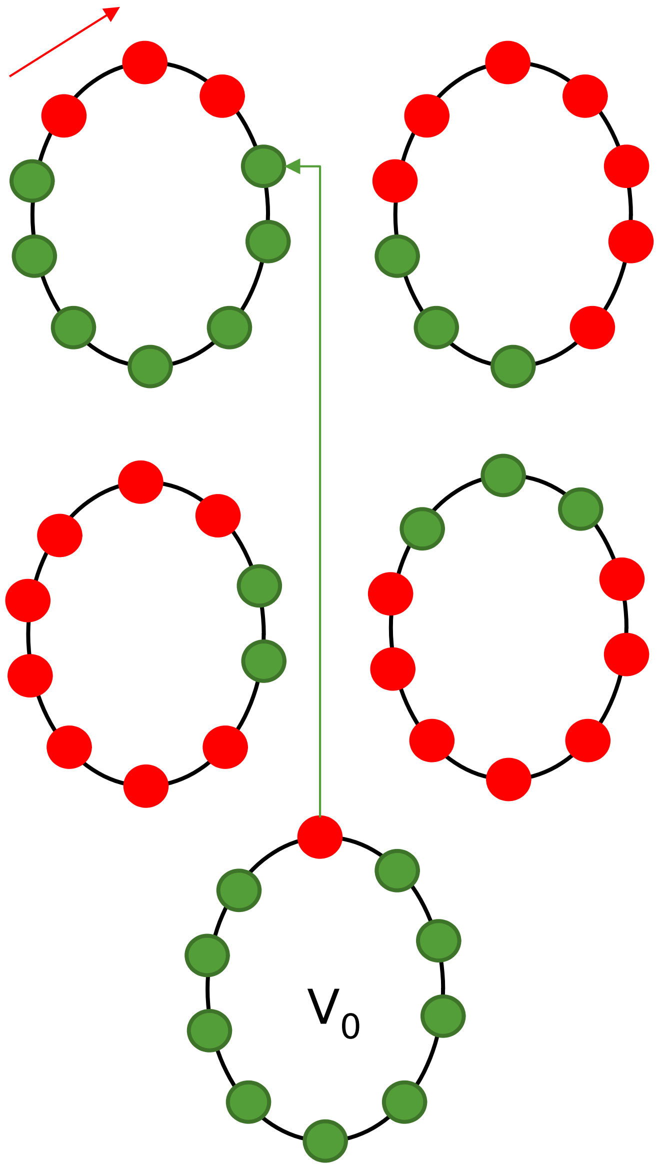
Continuous BGT

- Algorithm 3
  - Different partitions.
    - V_0: $h_i \leq n^{-2}$
    - V_1: $n^{-2} < h_i \leq 2n^{-2}$
    - V_2: $2n^{-2} < h_i \leq 4n^{-2}$
    - …
    - At most $\lceil 2\log_2 n \rceil$ sets



Continuous BGT

- Algorithm 3
  - Different partitions.
    - V_0: $h_i \le n^{-2}$
    - V_1: $n^{-2} < h_i \le 2n^{-2}$
    - V_2: $2n^{-2} < h_i \le 4n^{-2}$
    - …
    - At most $\lceil 2\log_2 n \rceil$ sets
- Upper bound: $O(\log_2 n\, h_{max}(V_i) \max\{D, MST(V_i)\})$
- Upper bound of V_0: $O\left(\frac{1}{n} D \log_2 n\right) = O(h_{max} D \log_2 n)$
- Approximation ratio: $O(\log_2 n)$



The main results

Algorithms	Approximation ratio	Time complexity in #steps
Discrete BGT		
Reduce-Max (Greedy approach)	$O(\log_2 n)$	$O(n \log n)$ + each round in $O(\log^2 n)$
Reduce-Fastest	4	$O(n \log n)$ + each round in $O(\log n)$
Reduce-to-Pinwheel	2 1+δ, balanced instances	$O(n)$ + each round in $O(1)$ $O(n \log n)$ + each round in $O(\log n)$
Continuous BGT		
Algorithm 1	$O(\frac{h_1}{h_n})$	Generation of MST $O(n^2)$ + each round in $O(1)$
Algorithm 2	$O(\log_2 \frac{h_1}{h_n})$	Partition $O(n)$ + Generation of MST $O(n^2)$
Algorithm 3	$O(\log_2 n)$	Partition $O(n)$ + Generation of MST $O(n^2)$

Note: We do not know whether BGT is tractable, similarly to closely related Pinwheel scheduling studied for years

Open Questions (Discrete BGT)

- Exact approximation ratios of *Reduce-Max* and *Reduce-Fastest*

  Bilò et al. (2021):

  - first constant upper bound of 9 for *Reduce-Max*

  - improved upper bound of ≈2.62 for *Reduce-Fastest*

  Simple examples: approximation ratios cannot be better than 2

- Improving the approximation ratio for arbitrary instances

  Van Ee (2021):

  - 2-approximation improved to 12/7-approximation

Open Questions (Discrete BGT)

- Designing efficient algorithms, easy to implement, avoiding reduction to Pinwheel

  Bilò et al. (2021):

  - 2-approximation

  - $O(n \log n)$ + each round in $O(\log n)$ amortized time

- Lower bounds on approximability
- Show hardness of discrete BGT problem + Pinwheel problem
- Improving the approximation ratio for balanced instances

Open Questions (Continuous BGT)

- Does Algorithm 3 have approximation ratio *o(log n)*?
- Is there any other algorithm with approximation ratio *o(log n)*?

  - discrete BGT and metric TSP (as two special cases of continuous BGT) are both approximable with approximation ratio O(1).
  - It is NP-hard to approximate continuous BGT with a factor better than 123/122 (as Metric TSP is a subproblem of continuous BGT).
  - approximation ratio for geometric / Euclidean BGT instances?
- Multiple gardeners