Open problems on optimal patrolling

Akitoshi Kawamura
(U Tokyo)

April 13, 2017

Fence patrolling

Patrolling
- Watch every point frequently
- A well-studied task in robotics
- Heuristics for various settings

Visit every point during any unit time

For example: patrolling a fence.
(line segment)

- *k* agents with speeds $v_1, \ldots, v_k$
- Maximize the fence's length

[CGKK] J. Czyzowicz, L. Gasieniec, A. Kosowski, and E. Kranakis, ESA 2011

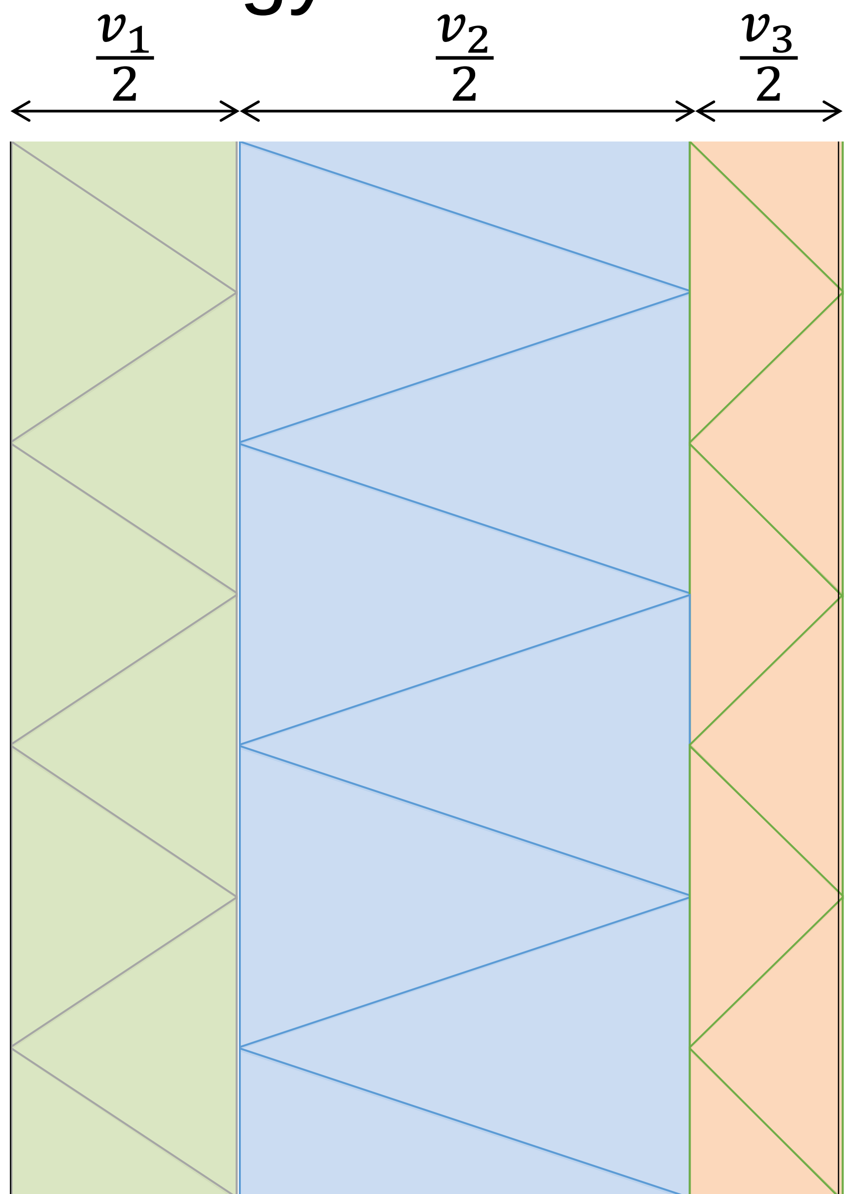
The Partition-Based Strategy

Divide the fence into parts, proportionally to the speeds

Total length $\frac{v_1+\cdots+v_k}{2}$

Question [CGKK]

Is PBS optimal?



[CGKK] J. Czyzowicz, L. Gasieniec, A. Kosowski, and E. Kranakis, ESA 2011

Various problem settings…

- Terrain
  - line segment, circle, trees, general graphs, …
  - sometimes we only need to protect part of the terrain (vertices)
  - patrolling a point (with the constraint that each robot can revisit it only after some predefined time) can be already interesting
- Idle time
  - Constant/different for different points
- Speed
  - Constant/different for different agents
- Objective
  - decision / optimization / approximation
    (maximize the profit, minimize the cost, …)

… with various applications in mind

Related Work. Patrolling has been intensely studied in robotics, especially in the last 4-5 years (cf. [3, 9–11, 14, 16, 21]). It is often viewed as a version of *coverage*, a central task in robotics. It is defined as the act of surveillance consisting in walking around an area in order to protect or supervise it. Patrolling is useful, e.g., to determine objects or humans that need to be rescued from a disaster environment. Network administrators may use mobile agent patrols to detect network failures or to discover web pages which need to be indexed by search engines, cf. [16]. Patrolling is usually defined as a perpetual process performed in a static or in a dynamically changing environment.

Notwithstanding several interesting applications and its scientific interest, the problem of boundary and area patrolling has been studied very recently (cf

J. Czyzowicz, L. Gasieniec, A. Kosowski, and E. Kranakis. Boundary Patrolling by Mobile Agents with Distinct Maximal Speeds. ESA 2011

The multi-item replenishment application. Another application of our model is to the problem of multi-item replenishment of m items that was first considered in Anily et al. (1998). In this problem, at each time slot, the stock of at most M of the items may be replenished. The costs involved are item-specific linear holding costs that are incurred at the end of each time slot and item-specific ordering costs that are incurred in the time slot in which the stock of the item is replenished. Also associated with each item is a demand per time slot that is the rate at which the item is consumed. It is required to have enough inventory of each item to meet the demand before its next replenishment. For item i, let d_i be its demand per time slot, c_i be its ordering cost, and h_i be its unit holding cost per time slot. Let $a_i = d_i h_i$. The holding cost for item i, j time slots before its next replenishment is therefore ja_i. The problem is to find an optimal policy specifying at which time slots to replenish stocks of each of the items so as to minimize the long-run average cost per time slot. This problem is modeled by GMPS with $b = 0$. Many variants of this problem are

A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. Mathematics of Operations Research 27(3):518-544, 2002.

Surveillance camera scheduling: Consider a system of robots carrying surveillance cameras, which patrol an area periodically [16]. Each robot has a predefined path that it needs to patrol, while recording all the events along this path. Upon completion of each patrol, the robot returns to the controller, where the recorded data is downloaded/processed, and the robot is prepared for its next tour. Each patrol j is associated with a revenue b_j, that is gained once the corresponding robot completes the tour. The controller can handle at most m robots simultaneously, and it takes p_j time units to complete the processing of robot j. The time it takes robot j to traverse its path is $t_j \geq 1$. The goal of the controller is to process the robots in a way that maximizes the profit gained from all robots, throughout the operation of the system. This yields an instance of SWV, where job j has a processing time p_j, and a window $a_j = p_j + t_j$.

Commercial broadcast: In a broadcasting system which transmits data (e.g., commercials on a running banner) there is some profit associated with the transmission of each commercial. This profit is gained only if some predefined period of time has elapsed since the previous transmission. The goal is to broadcast the commercials in a way that maximizes the overall profit of the system. Thus, we get an instance of SWV, where the window of each job corresponds to the time interval between consecutive transmissions of each commercial.

J. Sgall, H. Shachnai, and T. Tamir. Periodic scheduling with obligatory vacations. *Theoretical Computer Science* 410, 5112–5121, 2009.

Prologue

During his reign in the years 768–814, Charlemagne traveled constantly through his empire in Western Europe. Counts had been appointed to govern different pieces of Charlemagne's empire (called counties). On his travels, Charlemagne visited his counts regularly. One reason for these visits was to ensure the loyalty of his counts. Indeed, when a count was not visited for a certain period, the count would no longer obey Charlemagne, and declare independence, thereby rising against the emperor. Clearly, this would force Charlemagne to act and start an expensive war against the rebelling count. Charlemagne's challenge was to find a visiting sequence of his counts so that the time elapsed between two consecutive visits to a count would not exceed the "loyalty period" of that count.

1. Introduction

In the periodic latency problem, we are given a set of customers that need to be visited periodically. There is a

S. Coene, F. C. R. Spieksma, G. J. Woeginger. Charlemagne's challenge: The periodic latency problem. *Operations Research* 59(3):674-683, 2011.

Application in hardware design:

耐放射線性を有するアプリケーション特化型集積回路 (ASIC) の高位設計の研究過程で生じた最適化問題について議論する．自然界に存在する放射線の中で半導体メーカーが問題とする放射線は，アルファ線と中性子線である．これらの放射線はシリコンダイが維持している電位を破壊する．その結果，レジスタが記憶していた論理値が反転し ASIC が誤動作することがある．この問題の対策手法の一つとしてデータスクラビング (data scrubbing) がある．データスクラビングでは，レジスタを検査アクセスした際，訂正可能な 1 ビットエラーのデータが見つかると，訂正回路によって正しいデータに訂正しレジスタに書き戻す．この設計の基礎問題として，演算スケジュール済のデータフローグラフが回路仕様として与えられたとき，どのタイミングでどのレジスタにデータスクラビングを適用すれば訂正回路数最小を保証する耐放射線 ASIC が合成可能であるかという問題があり，区間の最大長指定分割問題として定式化可能である．この問題の入力は複数の区間の集合と最大区間長であり，出力はどの区間をいつ分割するかという分割指定である．最適化目標は，各時刻での分割数の最大値の最小化であり，これはデータスクラビングを実装するハードウェアコストの最小化と等しい．……

K. Inoue and M. Kaneko, On the decomposition problem of interval graphs with maximum length constraint, *IPSJ SIG Technical Report*, Vol.2016-AL-158 (情報処理学会アルゴリズム研究会), No. 19, 2016 (In Japanese).

Are the simple strategies optimal?

- For many problem settings, there is a natural strategy:
  - such as the partition-based strategy for line segment patrolling.
- But they are only optimal in some special settings.

Patrolling a fence

Two agents

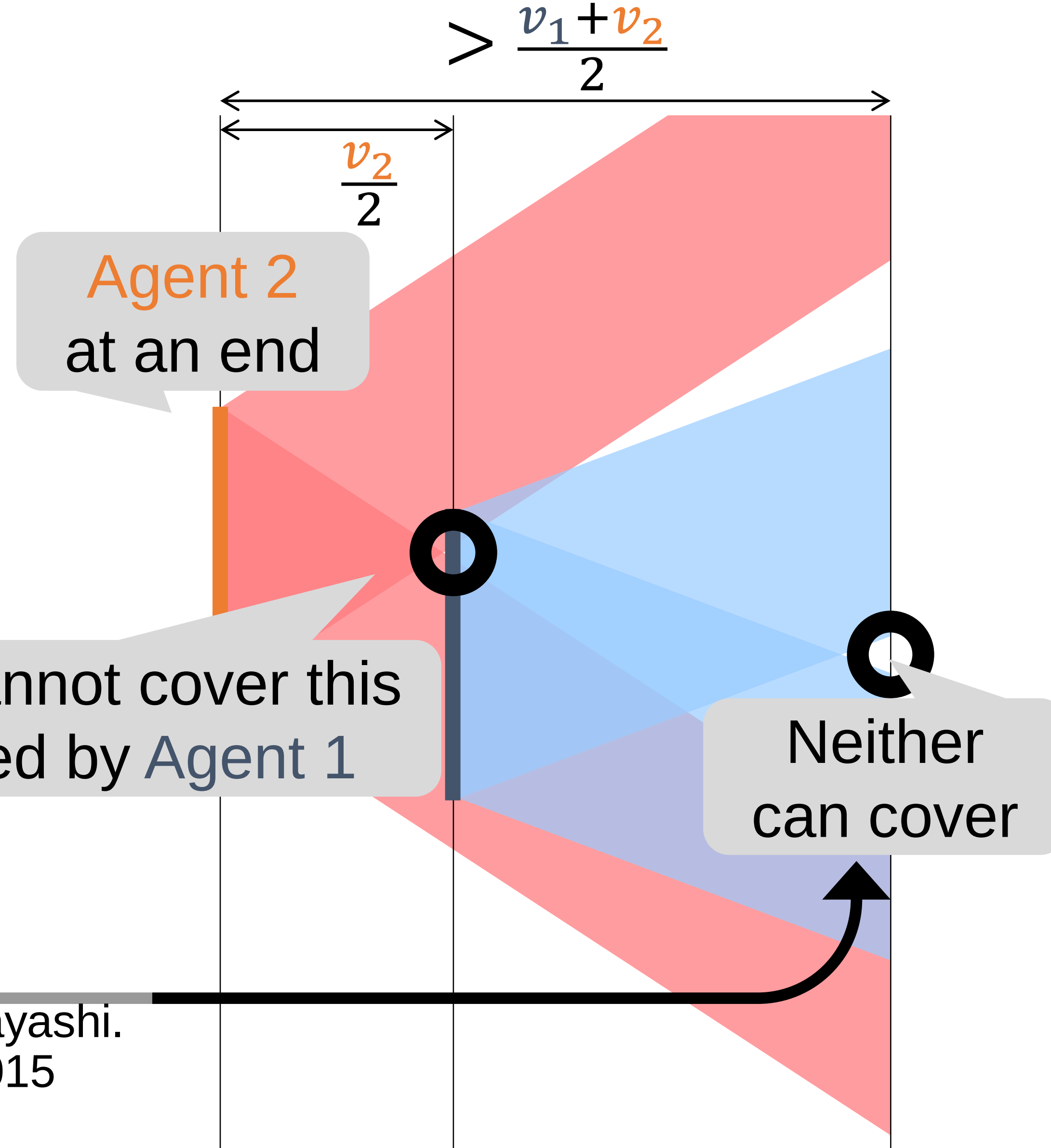
Theorem

PBS is optimal for two agents.

(speeds $v_1 \geq v_2$)

Theorem [KK]

PBS also optimal for three agents.



[KK] A. Kawamura and Y. Kobayashi. *Distributed Computing* 28, 2015

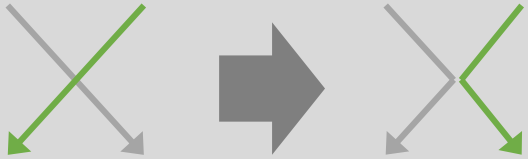
Same speed

Theorem

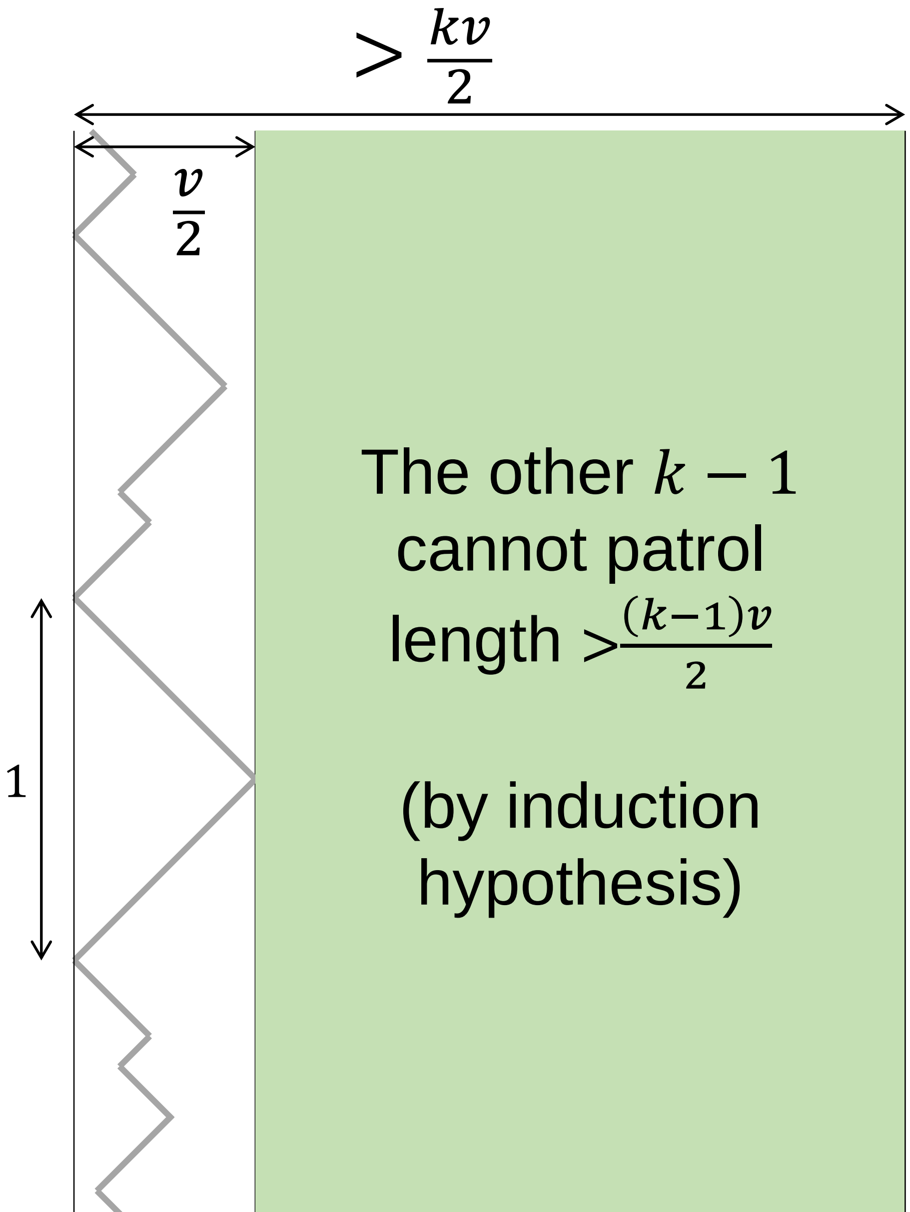
If all agents have one speed, PBS is optimal.

I.e. length $>\frac{kv}{2}$ cannot be patrolled.

We may assume that there is no switching:



➡ Left end is always covered by one agent



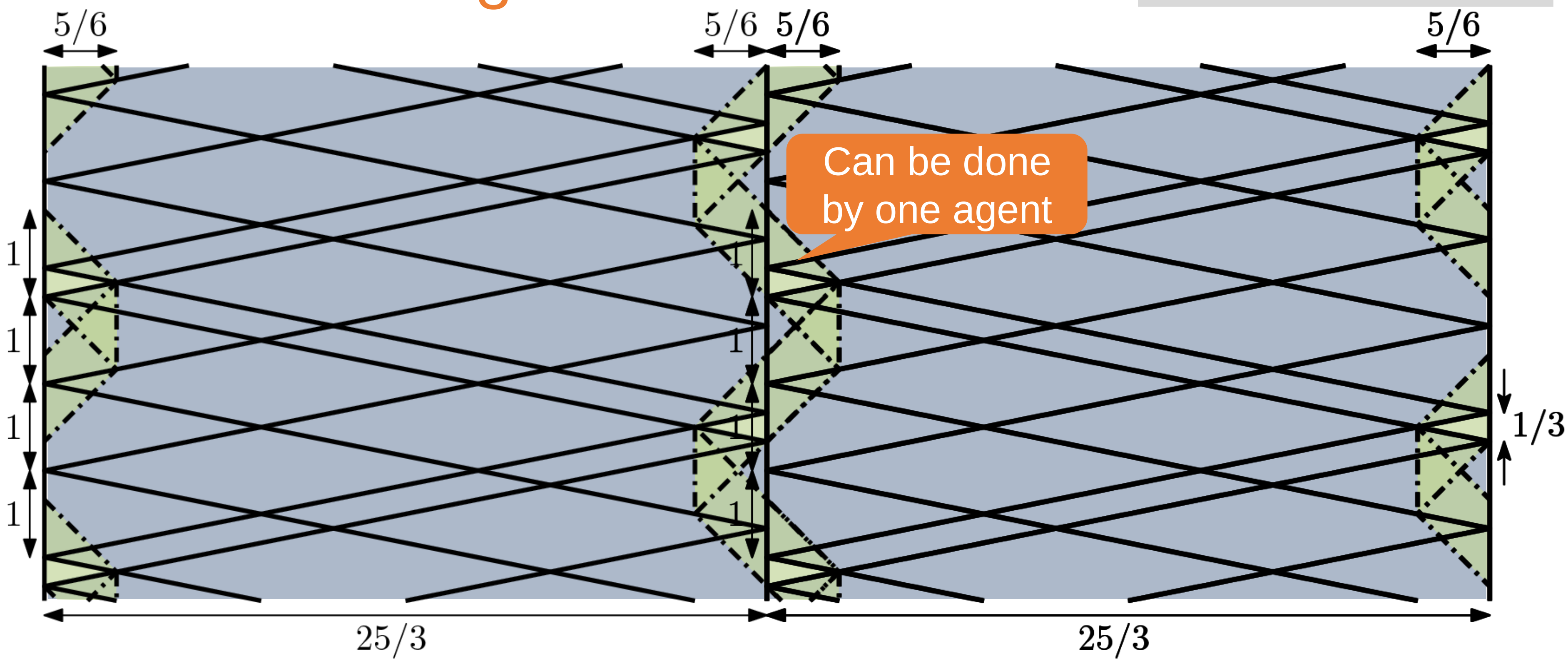
PBS is not always optimal

6 x speed 5

~~4~~ 3 x speed 1

=

PBS would patrol $\frac{\cancel{34}\ 33}{2}$



Upper/lower bounds

What is the largest c such that
the following is true for all $v_1, \ldots, v_k$?

> No fence of length $> c(v_1 + \cdots + v_k)$ can be patrolled by agents with speeds $v_1, \ldots, v_k$.

$c \leq 1$ (by the area argument)

Conjecture:
This is the best

$c \geq 0.666\ldots$ [KS]

$c \geq 0.520\ldots$ [DGT]

$c \geq 0.512\ldots$ [KK]

Question:
Is this the best?

$c \geq 0.5$ (by PBS)

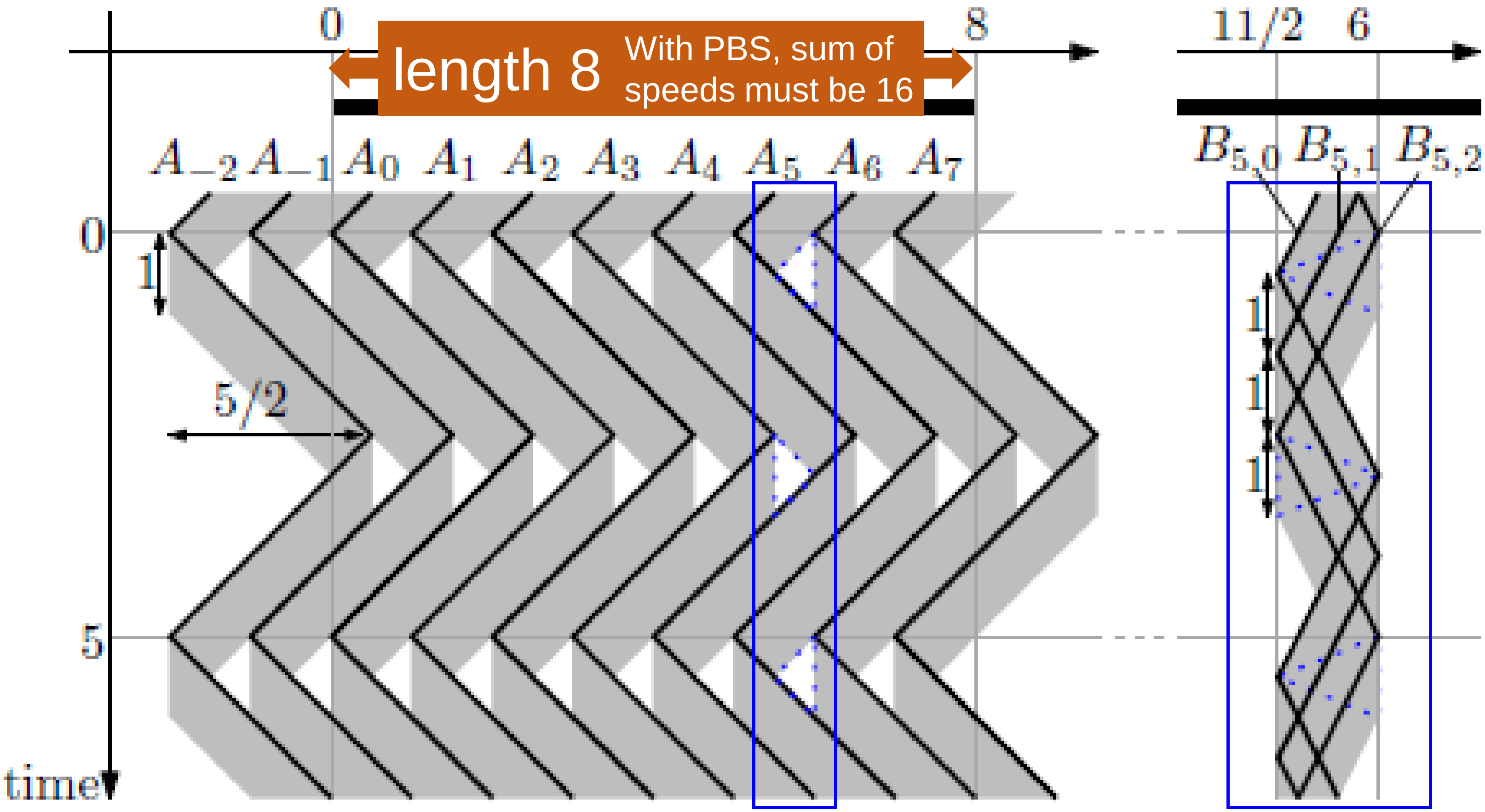
[KS] A. Kawamura and M. Soejima, CIAC 2015

[DGT] A. Dumitrescu, A. Ghosh, and C.D. Tóth. *Electronic Journal of Combinatorics* 21, P3.4, 2014.

[KK] A. Kawamura and Y. Kobayashi. *Distributed Computing* 28, 2015

Generalizing this, we can patrol fences of length $\left(\frac{2}{3} - \varepsilon\right) \sum_i v_i$

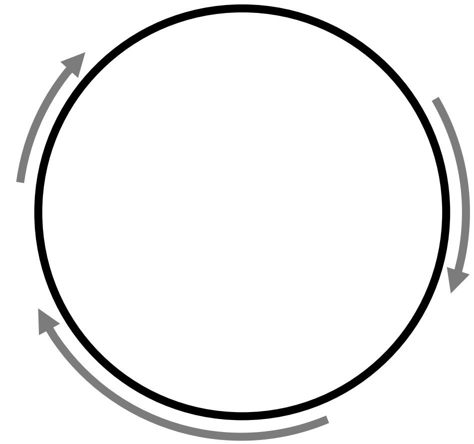
A schedule that patrols more than PBS



10 with speed 1 + 3 with speed $\frac{1}{5}$ × 8

= 14.8

Patrolling a cycle



This can really make a difference [CGKK]

- Speeds $v_1 \geq \cdots \geq v_k$
- Agents move clockwise only
- Every point must be visited during any unit time
- Want to maximize the perimeter

Simple strategy: The fastest r agents move at speed v_r

➡ Perimeter $\max_r r v_r$

Conjecture [CGKK]	**Conjecture** [DGT]
This is optimal.	It is a constant-ratio approximation.

No [DGT, KS].

No [in preparation].

[CGKK] J. Czyzowicz, L. Gasieniec, A. Kosowski, and E. Kranakis, ESA 2011

[DGT] A. Dumitrescu, A. Ghosh, and C.D. Tóth. *Electronic Journal of Combinatorics* 21, 2014

[KS] A. Kawamura and M. Soejima, CIAC 2015

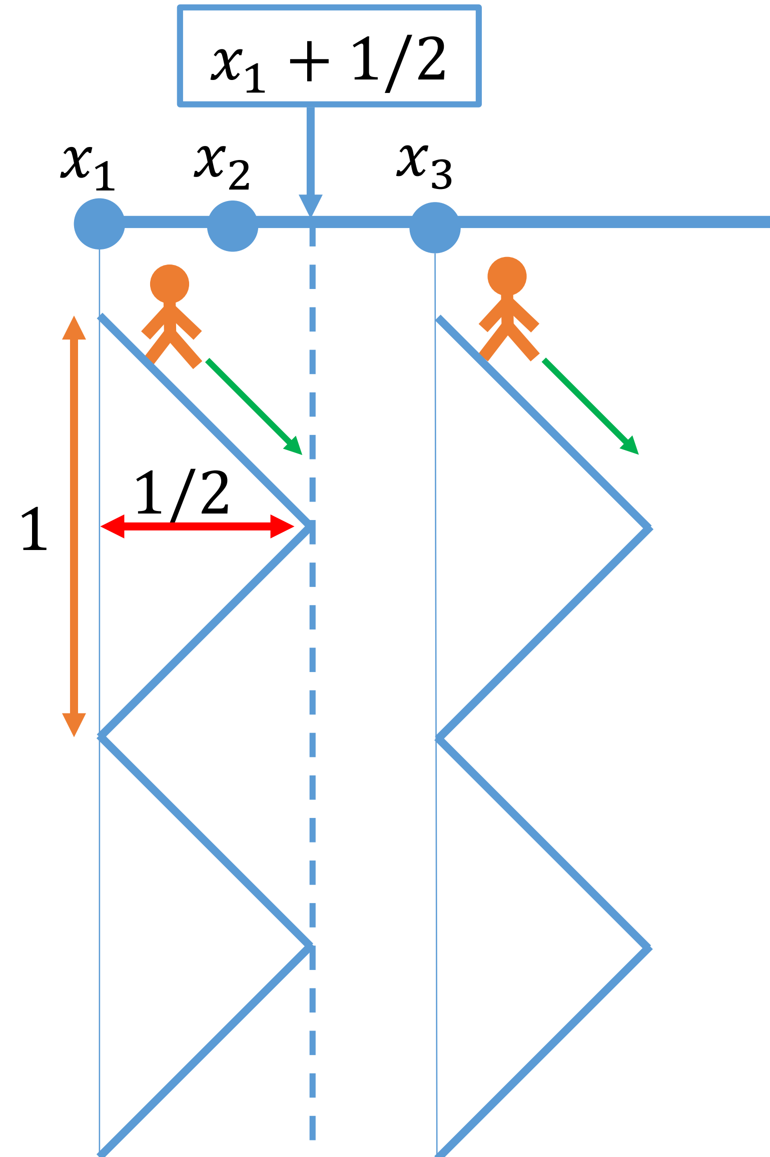
Patrolling vertices

Patrolling on a graph (path)

What if we want to guard
not all points on the fence,
but just a specified set of *vertices*?

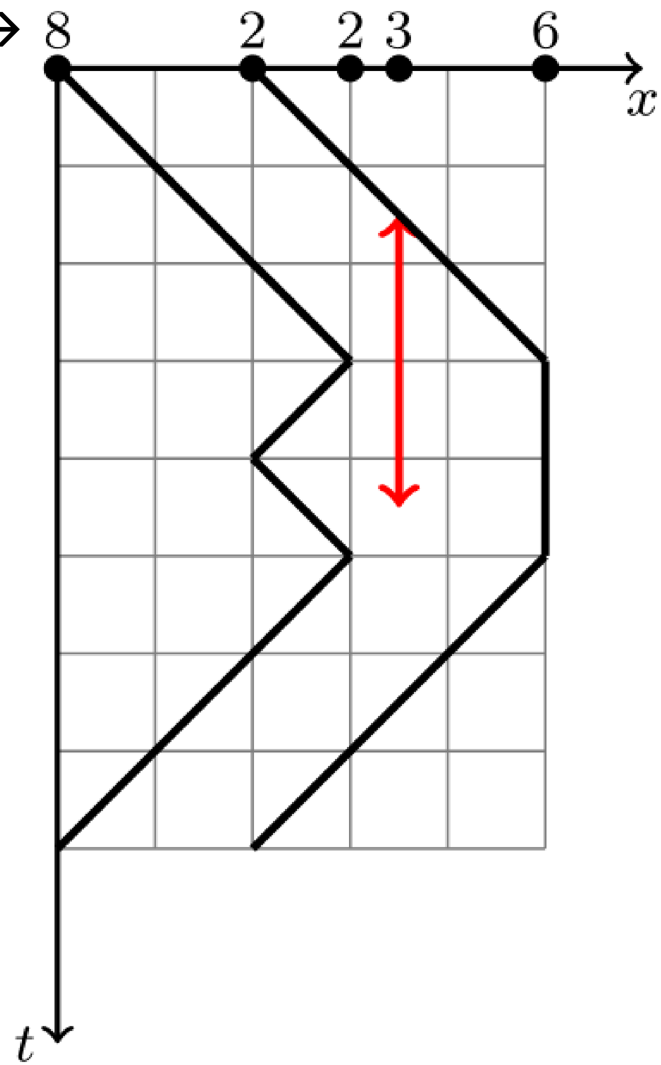
For same-speed agents,
PBS-like strategy remains optimal.

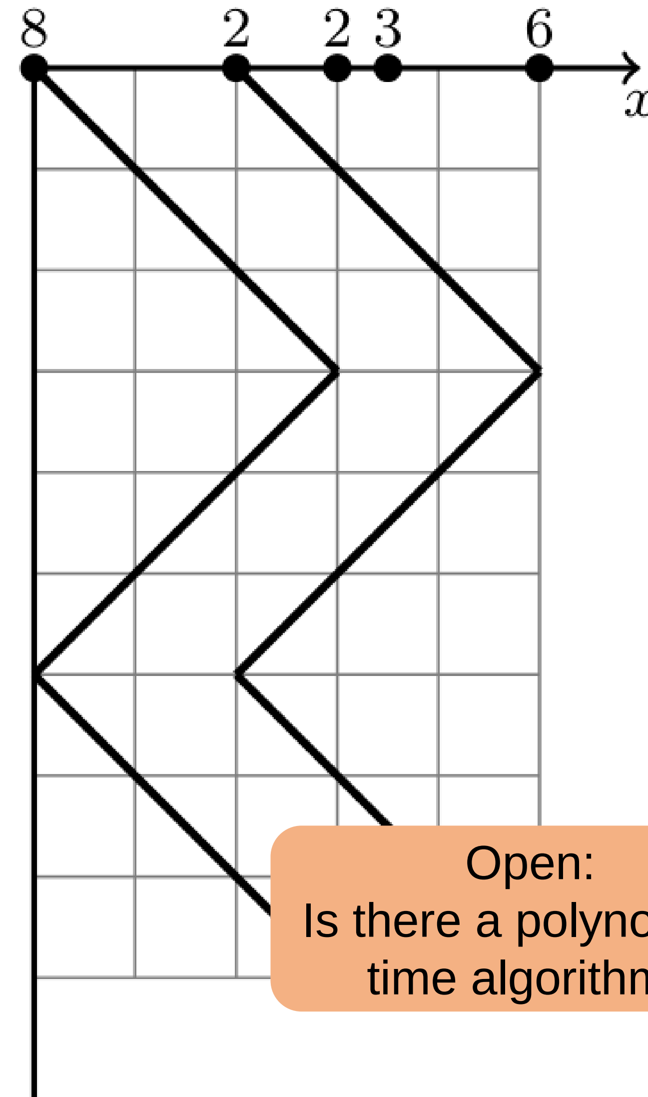
However…



When the vertices have different idle times, the simple strategy (greedily determining the movement from the left) does not work.

idle times →



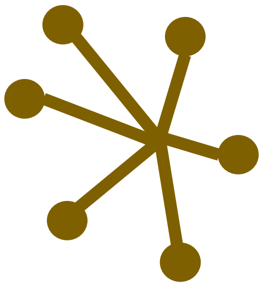


Open:
Is there a polynomial-time algorithm?

One agent for each vertex

If we require that each vertex be guarded by one agent, then the problem for paths and same-speed agents can be solved easily (even with different idle times) [CSW].

But not for more general graphs:
For example, it is NP-hard for stars, even for uniform idle time [CSW].



Surprisingly, the problem for stars with uniform idle time can be solved easily if we remove the requirement [KN].

(next slide)

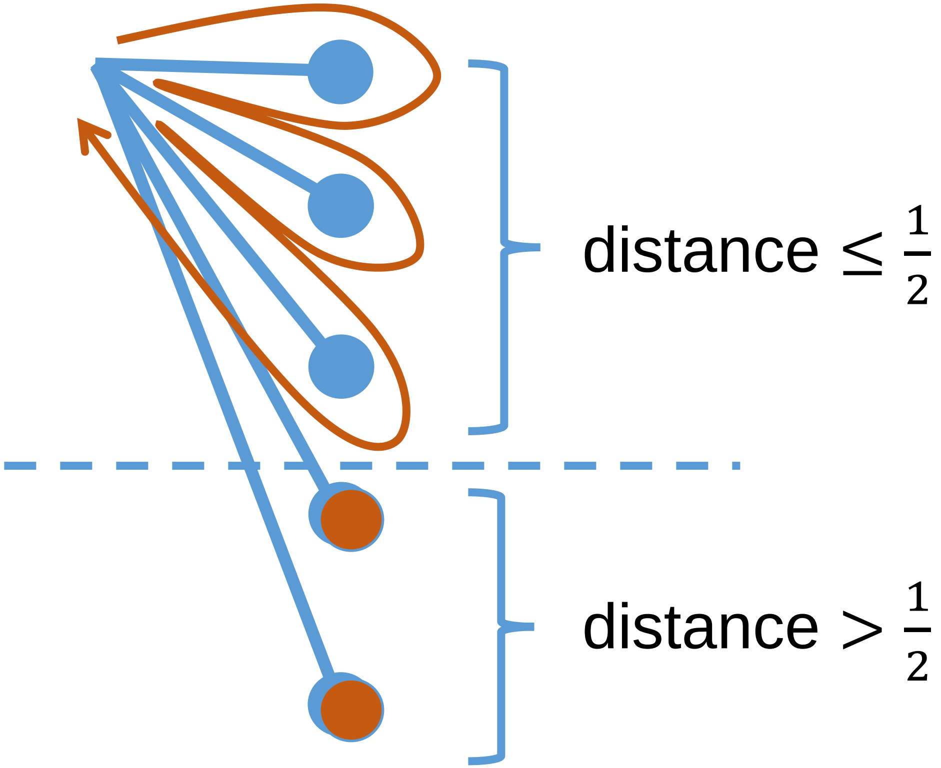
[CSW] S. Coene, F.C.R. Spieksma, and G.J. Woeginger. Charlemagne's challenge: the periodic latency problem. *Operations Research* 59(3), 674–683, 2011.

[KN] 河村、能城「複数の巡査の協力による指定地点の警邏について」電子情報通信学会総合大会2017.

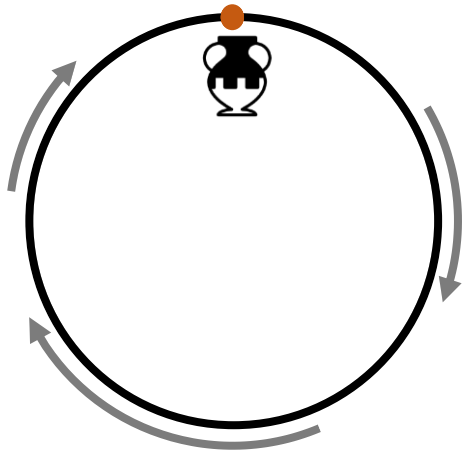
Stars, idle time 1, cooperative

Assign an agent to each vertex
at distance $> 1/2$ from the root.

For other vertices,
the remaining agents
move cyclically
with time gap 1.



Point patrolling



What if we only need to guard one point on the cycle?

Let's say an agent can only watch the point when it arrives at it (i.e., staying does not help).

Agent i can visit the point only once in $a_i \coloneqq \frac{\text{perimeter}}{\text{speed of } i}$.

Thus,

Point patrolling problem

Given $a_1, \ldots, a_k > 0$, is there a schedule where

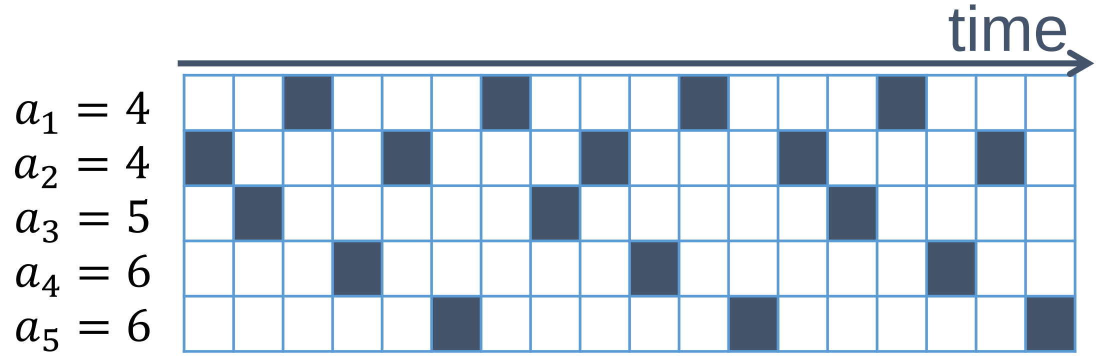
- agent i never visits the point twice during any time period of length a_i; and
- the point is visited by some agent during any time period of length 1.

Note:

- This is a decision problem; to maximize perimeter, use binary search.
- Time can be discretized (round a_i up to the nearest integer).

Patrolling a point

Point patrolling



- For each agent i, the minimum gap $a_i \in \mathbf{N}$ is specified. (After a visit, it has to wait for a_i before its next visit.)
- Every integer time must be visited by some agent.

$(4, 4, 5, 6, 6)$ ➡ **Yes**

$(a_1, \dots, a_k)$ ➡ **?** Conjecture: NP-hard

$$\frac{1}{a_1} + \cdots + \frac{1}{a_k} < 1$$ ➡ **No**

$$\frac{1}{a_1} + \cdots + \frac{1}{a_k} \geq 1.546^{\text{[KS]}}$$ ➡ **Yes**

Conjecture: 1.264

$(2, 3, 4)$ ➡ **Yes**

$\underline{\left(2, 3, 5, 9, \dots, 2^k + 1\right)}$ ➡ **No**

Is it even in $\mathbf{NP}$?

[KS] A. Kawamura and M. Soejima, CIAC 2015

Theorem

If there is a schedule for a given set of agents, there is a periodic schedule with period $\le a_1 a_2 \cdots a_k$.

Proof sketch

Given a schedule, let $s_{t,i} \in \{0, \dots, a_i - 1\}$ be the time that agent i at time t needs to wait until its next visit.

There are only $a_1 a_2 \cdots a_k$ possible sequences $S_t = (s_{t,1}, \dots, s_{t,k})$.

If $S_t = S_{t'}$, repeat the schedule during time $t, \dots, t' - 1$.

This implies that the problem is in $\mathbf{NEXP}$.

Question:

Is it in $\mathbf{NP}$?

Constant-gap restriction

What if we require that agent i must make a visit *exactly* at time a_i after its previous visit? In other words:

Periodic covering problem: Given $(a_1, \ldots, a_k)$, is there $(r_1, \ldots, r_k)$ such that the sets $\{ a_i n + r_i \mid n \in \mathbf{Z} \}$ cover $\mathbf{Z}$?

Conjecture: This is NP-hard

We can also consider the "dual" problem:

Periodic "packing" problem: Given $(a_1, \ldots, a_k)$, is there $(r_1, \ldots, r_k)$ such that the sets $\{ a_i n + r_i \mid n \in \mathbf{Z} \}$ are all disjoint?

Theorem: This is NP-hard (next slide)

(Note: This problem is in $\mathbf{NP}$.)

We conjecture that in fact these problems are NP-hard even when restricted to inputs $(a_1, \ldots, a_k)$ such that

$$\frac{1}{a_1} + \cdots + \frac{1}{a_k} = 1.$$

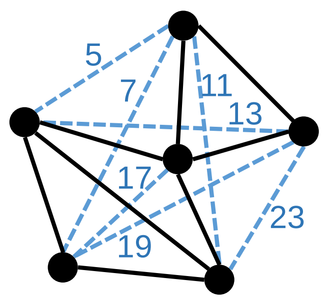
In this case,

- the covering and packing problems coincide; and
- the constant-gap restriction is implied automatically.

Theorem

The periodic packing problem is NP-complete.

Reduction from graph 3-colouring



$G = (\{1, \ldots, k\}, E)$ $\qquad$ $\overline{G} = \left(\{1, \ldots, k\}, \overline{E}\right)$

Assign distinct primes p_e for edges $e \in \overline{E}$

For each $i \in \{1, \ldots, k\}$, create an agent $a_i = 3 \prod_{e \in \overline{E}(i)} p_e$

where $\overline{E}(i)$ is the set of edges incident to i in $\overline{G}$

Then we have

G is 3-colourable $\Leftrightarrow$ packing possible with $a_1, \ldots, a_k$

because…

vertices i and j can be assigned the same colour in G

$\Leftrightarrow$ vertices i and j are adjacent in $\overline{G}$

$\Leftrightarrow$ numbers a_i and a_j have a common divisor besides 3

$\Leftrightarrow$ agents i and j can be assigned into the same 3-residue.

Further open questions

- Very simple problems remain open
  - Optimality of simple strategies
  - Approximation ratio
  - Hardness
- More practical settings
  - What if we have “regions” and the agents can “see”?
  - Distributed, fault tolerant, …