Approximation Algorithms
Chapter 5: Freeze Tag and Scan Cover

Sándor P. Fekete

Algorithms Division
Department of Computer Science
TU Braunschweig

1. Introduction
2. Freeze Tag
3. Angular Freeze Tag
4. Angular Scan Cover

1. Introduction

2. Freeze Tag

3. Angular Freeze Tag

4. Angular Scan Cover

Freeze Tag

Technische
Universität
Braunschweig

Freeze Tag

Algorithmica (2006) 46: 193–221
DOI: 10.1007/s00453-006-1206-1

Algorithmica

The Freeze-Tag Problem: How to Wake Up a Swarm of Robots[1]

Esther M. Arkin,[2] Michael A. Bender,[3] Sándor P. Fekete,[4]
Joseph S. B. Mitchell,[2] and Martin Skutella[5]

Abstract. An optimization problem that naturally arises in the study of swarm robotics is the *Freeze-Tag Problem* (*FTP*) of how to awaken a set of "asleep" robots, by having an awakened robot move to their locations. Once a robot is awake, it can assist in awakening other slumbering robots. The objective is to have all robots awake as early as possible. While the FTP bears some resemblance to problems from areas in combinatorial optimization such as routing, broadcasting, scheduling, and covering, its algorithmic characteristics are surprisingly different.

We consider both scenarios on graphs and in geometric environments. In graphs, robots sleep at vertices and there is a length function on the edges. Awake robots travel along edges, with time depending on edge length. For most scenarios, we consider the offline version of the problem, in which each awake robot knows the position of all other robots. We prove that the problem is NP-hard, even for the special case of star graphs. We also establish hardness of approximation, showing that it is NP-hard to obtain an approximation factor better than $\frac{5}{3}$, even for graphs of bounded degree.

These lower bounds are complemented with several positive algorithmic results, including:

- We show that the natural greedy strategy on star graphs has a tight worst-case performance of $\frac{7}{3}$ and give a polynomial-time approximation scheme (PTAS) for star graphs.
- We give a simple $O(\log \Delta)$-competitive online algorithm for graphs with maximum degree Δ and locally bounded edge weights.
- We give a PTAS, running in nearly linear time, for geometrically embedded instances.

The Freeze-Tag Problem: How to Wake Up a Swarm of Robots

Estie Arkin[1]
Michael Bender[1]
Sándor Fekete[2]
Joe Mitchell[1]
Martin Skutella[3]

[1] University at Stony Brook
[2] TU Braunschweig
[3] TU Berlin

Freeze Tag!:

Given: n robots at points in a metric space

$n-1$ robots are "asleep"

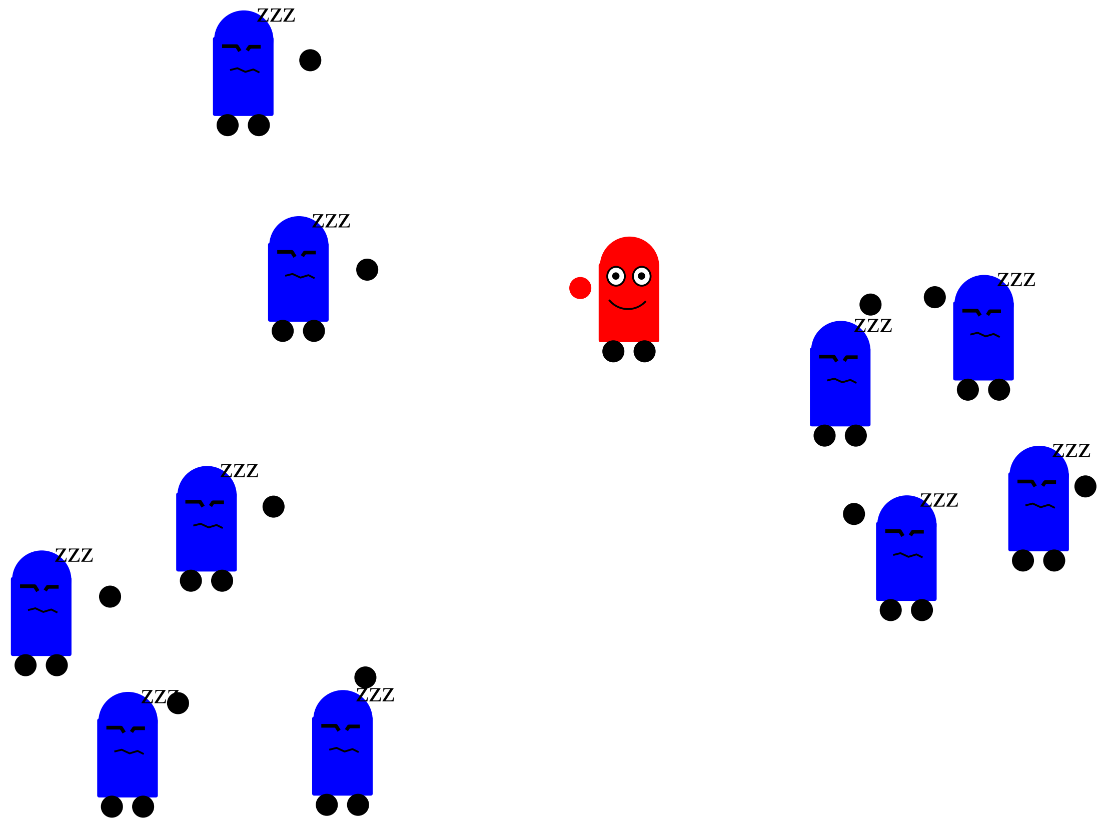
1 robot is awake

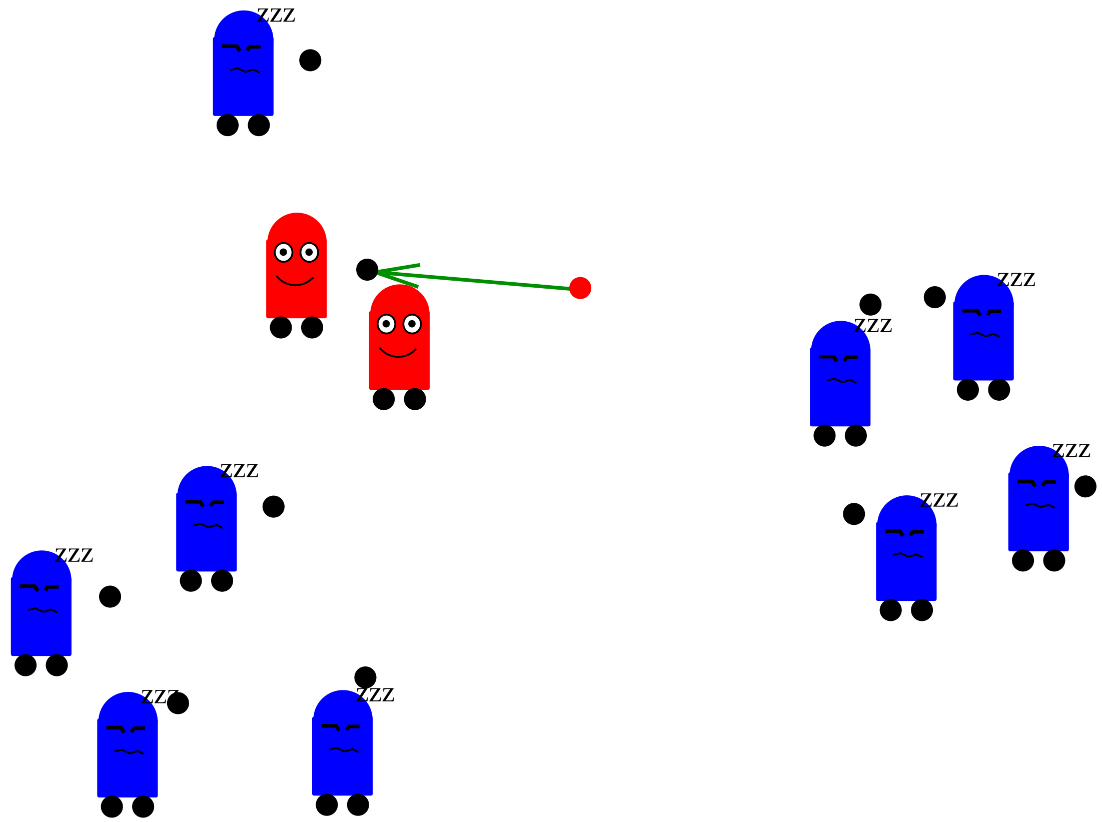
An awake robot "wakes up" a sleeping robot at point p by going to p

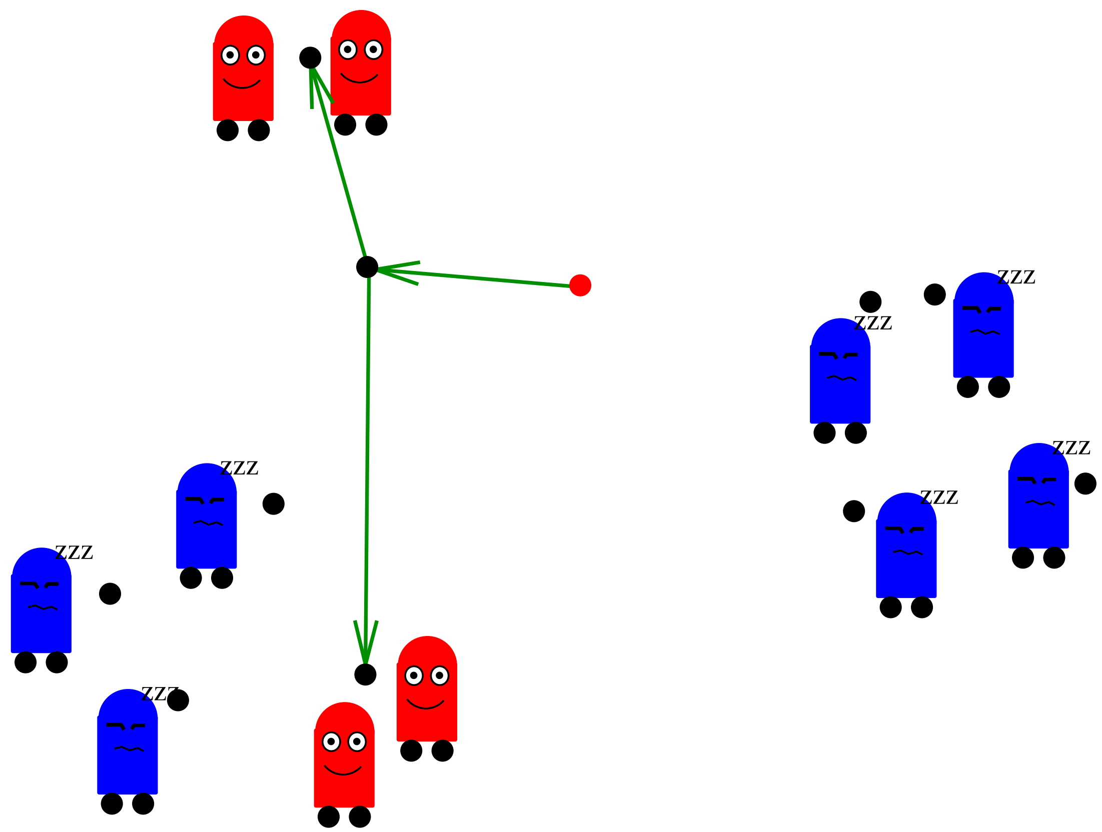
As robots wake up, there are more robots to assist in waking up others

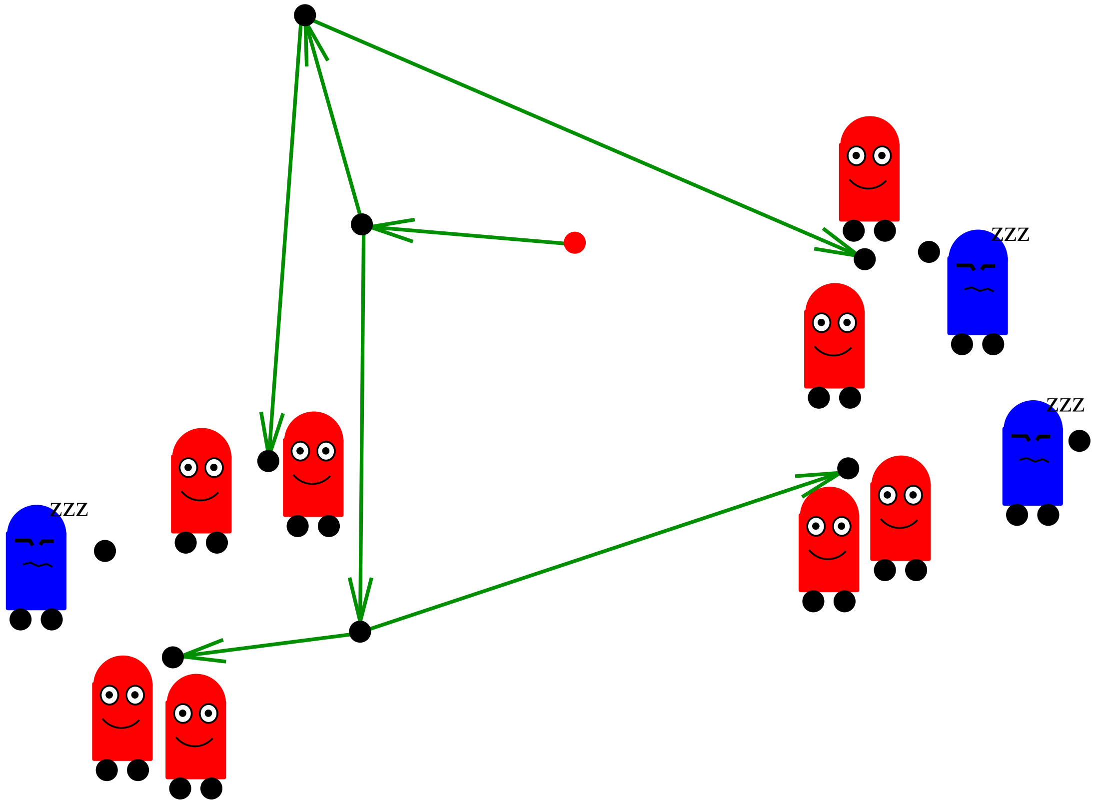
Goal: Wake up all robots as soon as possible

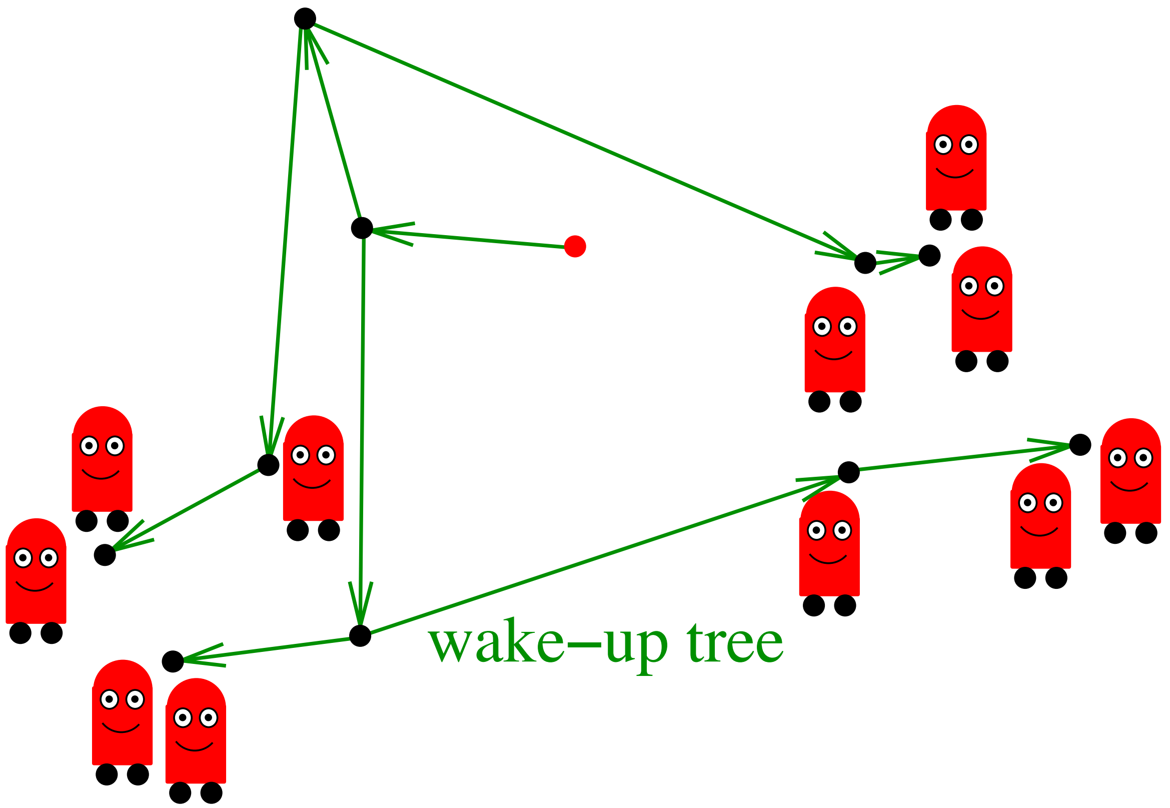
Minimize makespan


ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ


ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ


ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ
ZZZ


ZZZ
ZZZ
ZZZ


wake–up tree

Other Motivations:

• Distribute data (or other commodity) to agents, where physical proximity is necessary for transmittal
• Secret sharing by whisper
• Natural network optimization problem:

Minimize the length of a root-to-leaf path in a *binary* spanning tree

Related Work:

Dissemination of data in graphs:

- minimum broadcast time problem
- multicast problem
- minimum gossip time problem

Key differences from FTP:

- messages sent along edges of graph (no need for proximity)
- broadcast problem poly in trees, but FTP is NP-hard even for stars

Simple Approximation Bounds:

Any "brain-dead" strategy gives $O(\log n)$-approx:

- Source robot awakens one other robot

  (travelling distance $\leq D$, diameter)

  Now 2 awake robots.

- Each travels to a distinct other asleep robot (dist $\leq D$), awakens it and waits (if necessary) for the other robot to reach its destination

  Now 4 awake robots.

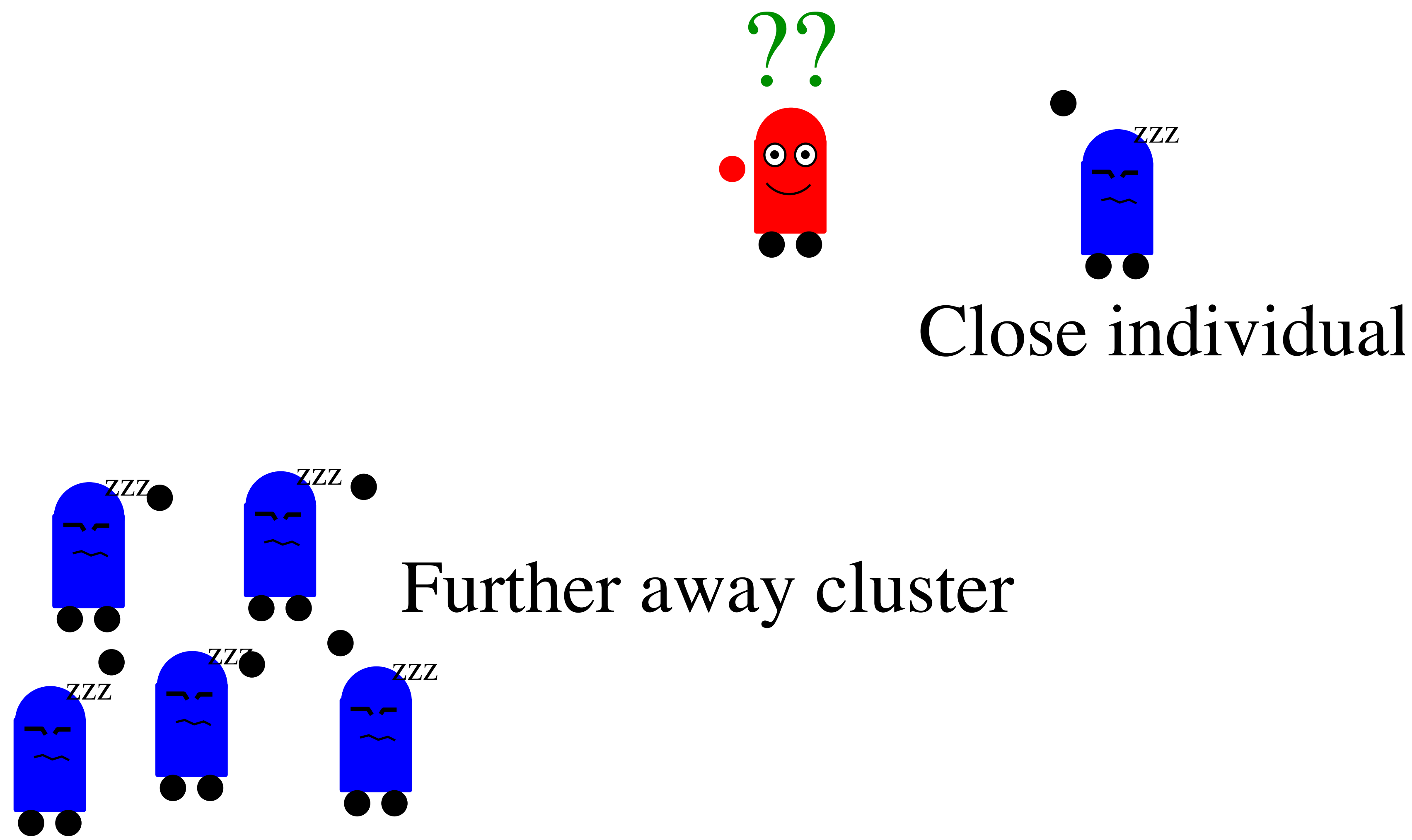
- Etc, etc, $\log n$ rounds, each of length $\leq D$

- Lower bound on OPT: $t^* \geq R_0 = \max_{p_i} d(v_0, p_i) \geq D/2$

OPEN: Is there an $o(\log n)$ approximation algorithm?

Fundamental Question:

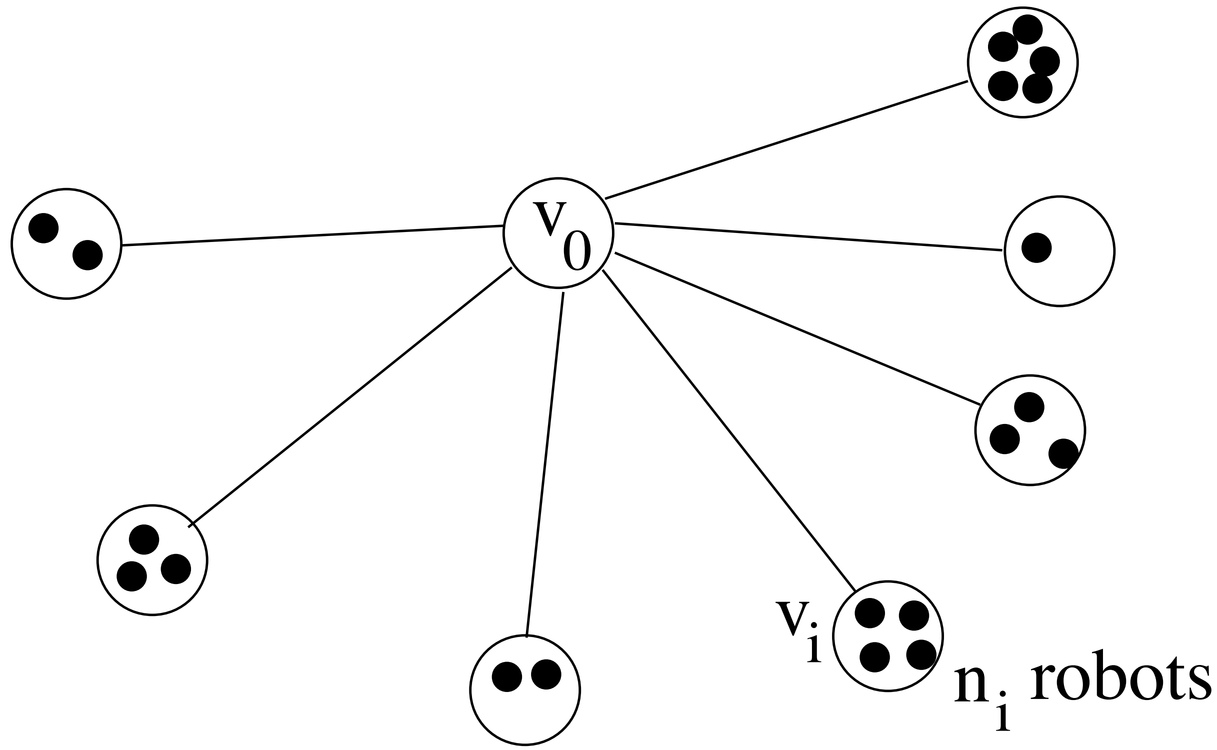
Whether to awaken a nearby robot or go further to (start to) awaken a larger swarm?



Summary of Results:

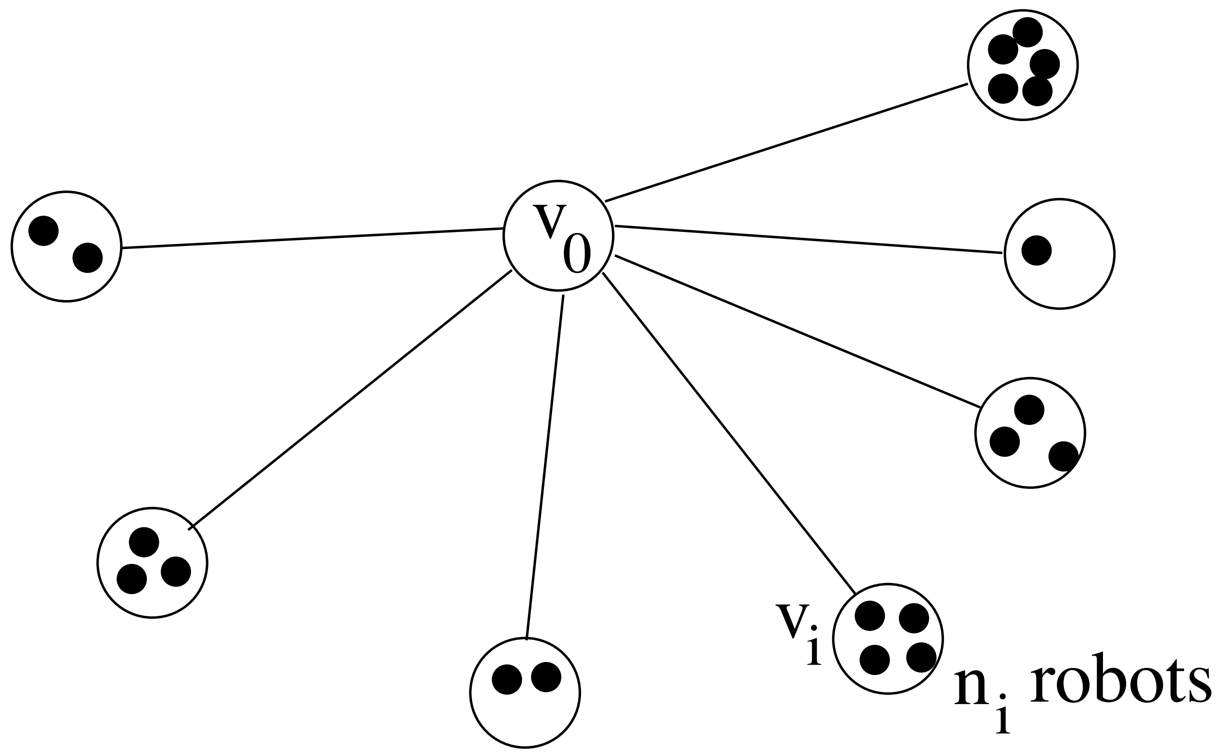
1. FTP is NP-hard, even for stars, with one robot per leaf
2. $O(1)$-approx for (general) stars
3. PTAS for stars, same number of robots at each leaf
4. Tight analysis of greedy heuristic on stars: 7/3-approx
5. $o(\log n)$-approx for FTP in ultrametrics ($2^{O(\sqrt{\log\log n})}$-approx)
6. Simple linear-time on-line algorithm, $O(\log \Delta)$-competitive
   ($\Delta \leq$ max degree)
7. NP-hard to get <5/3-approx in offline problem, even if $\Delta \leq 5$
8. PTAS for geometric instances in fixed-dimension, L_p metric
   Time $O(n \log n + 2^{poly(1/\varepsilon)})$

Stars: Equal-Length Spokes:



Natural greedy strategy:

A robot arriving to v_0 chooses the (unclaimed) leaf having the most asleep robots



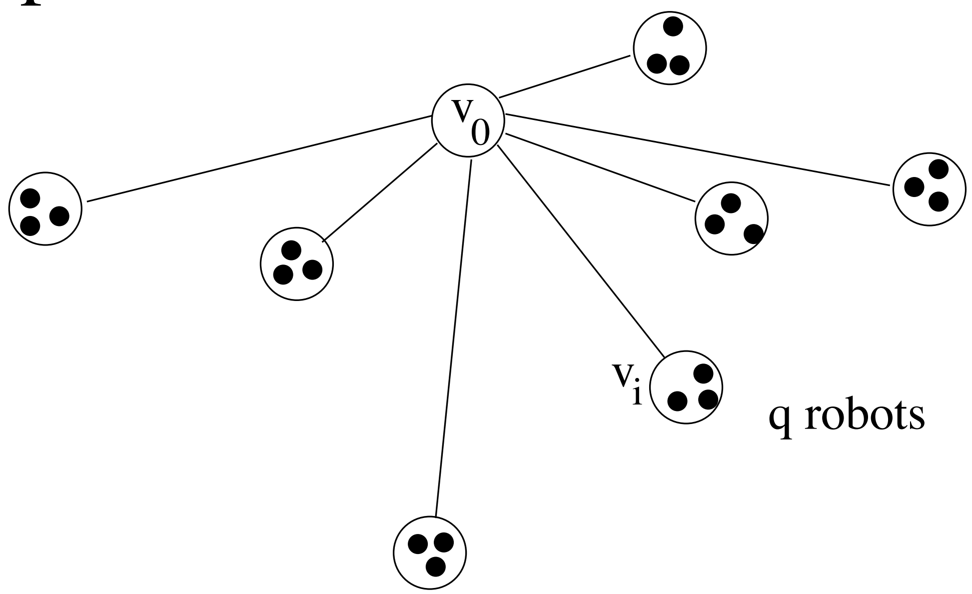
Claim: Greedy is optimal

Proof: two exchange arguments:

- $\exists$ optimal schedule in which no robot is idle if $\exists$ work to be done
- if a robot chooses a branch B with fewer robots than unclaimed branch B', swap

Stars: Unequal Length Spokes:

Assume q robots asleep at each of n leaves

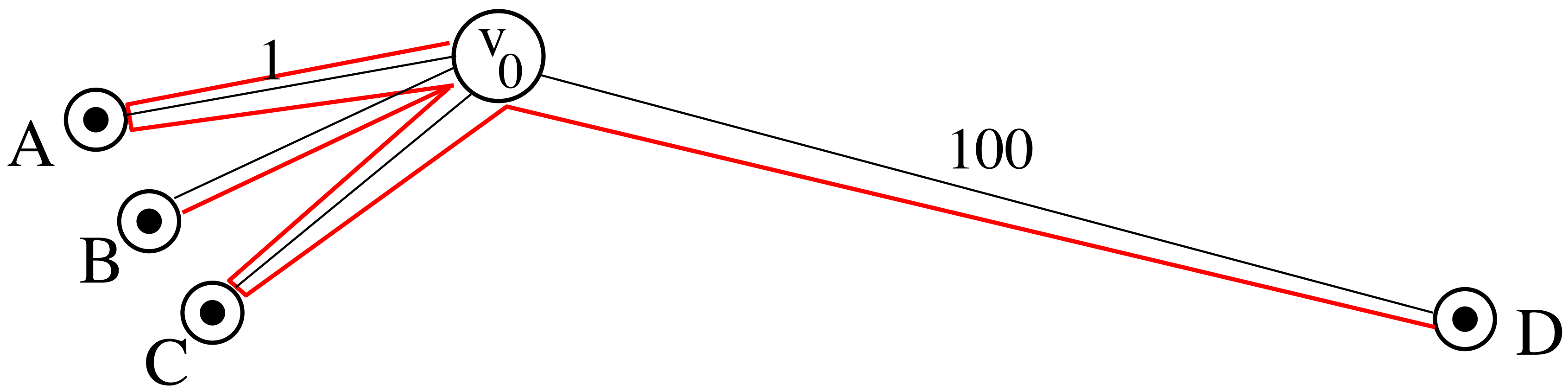


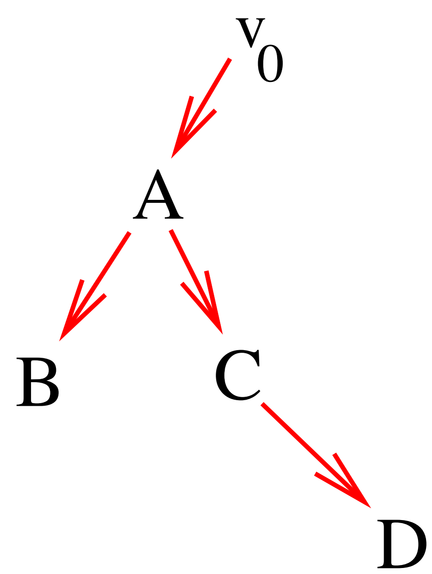
Natural greedy strategy:

> A robot arriving to v_0 chooses the *shortest* (unclaimed) spoke having asleep robots

Question: How good is Greedy in this case?

Example: Greedy Strategy:

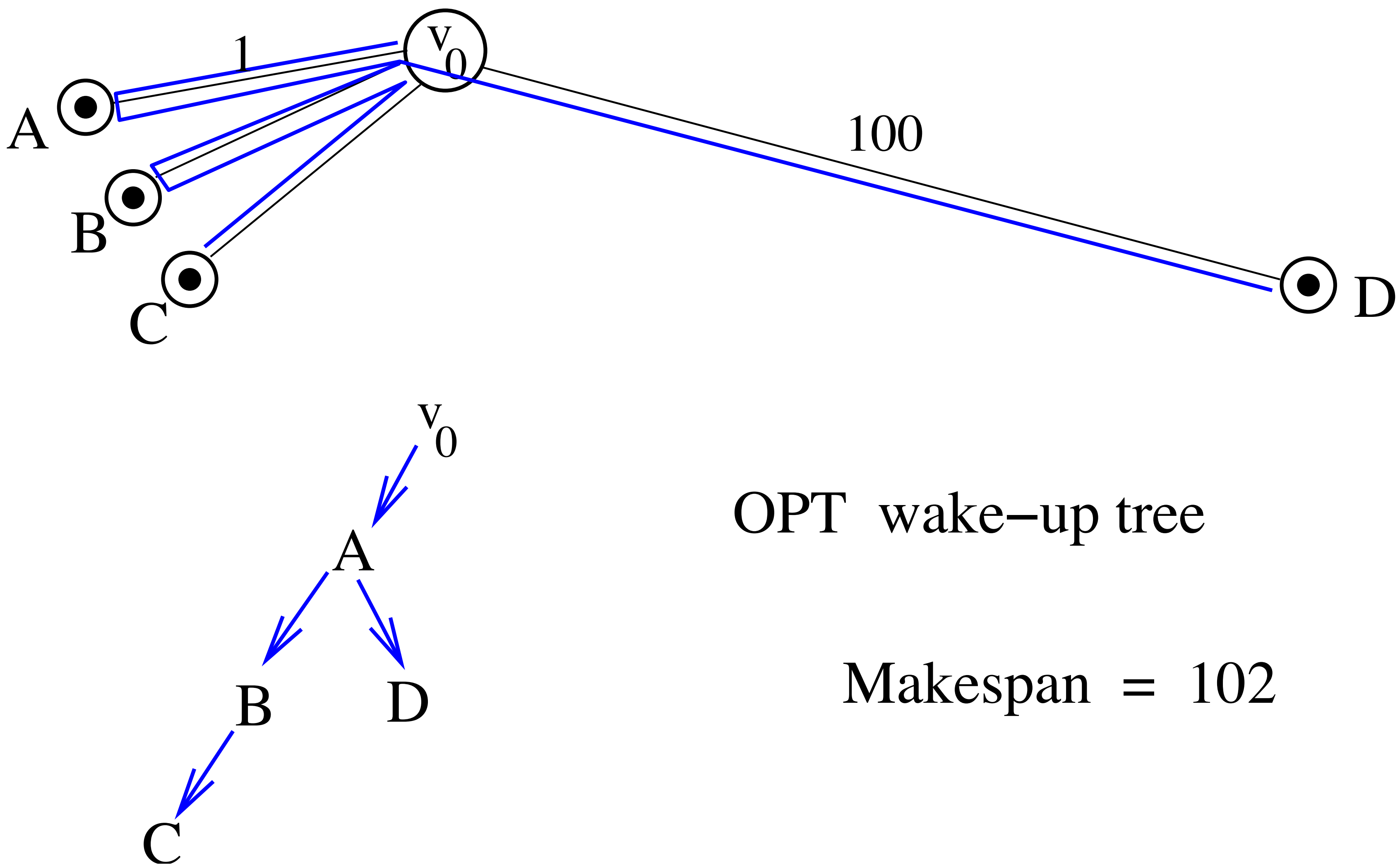




GREEDY wake−up tree

Makespan = 104

Example: Optimal Strategy:



OPT wake–up tree

Makespan = 102

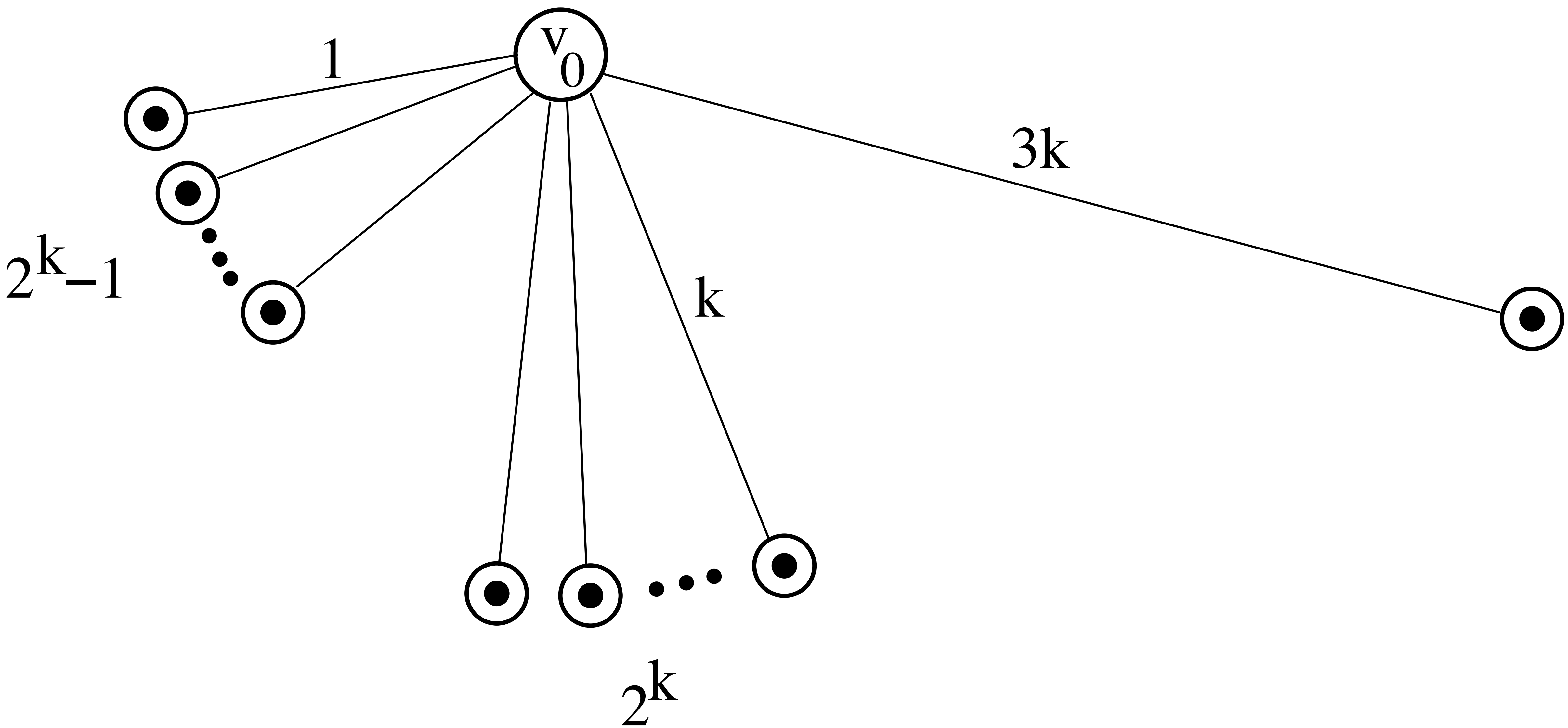
Thus, Greedy can be *suboptimal* (104 vs. 102)

Analysis of Greedy on Stars:

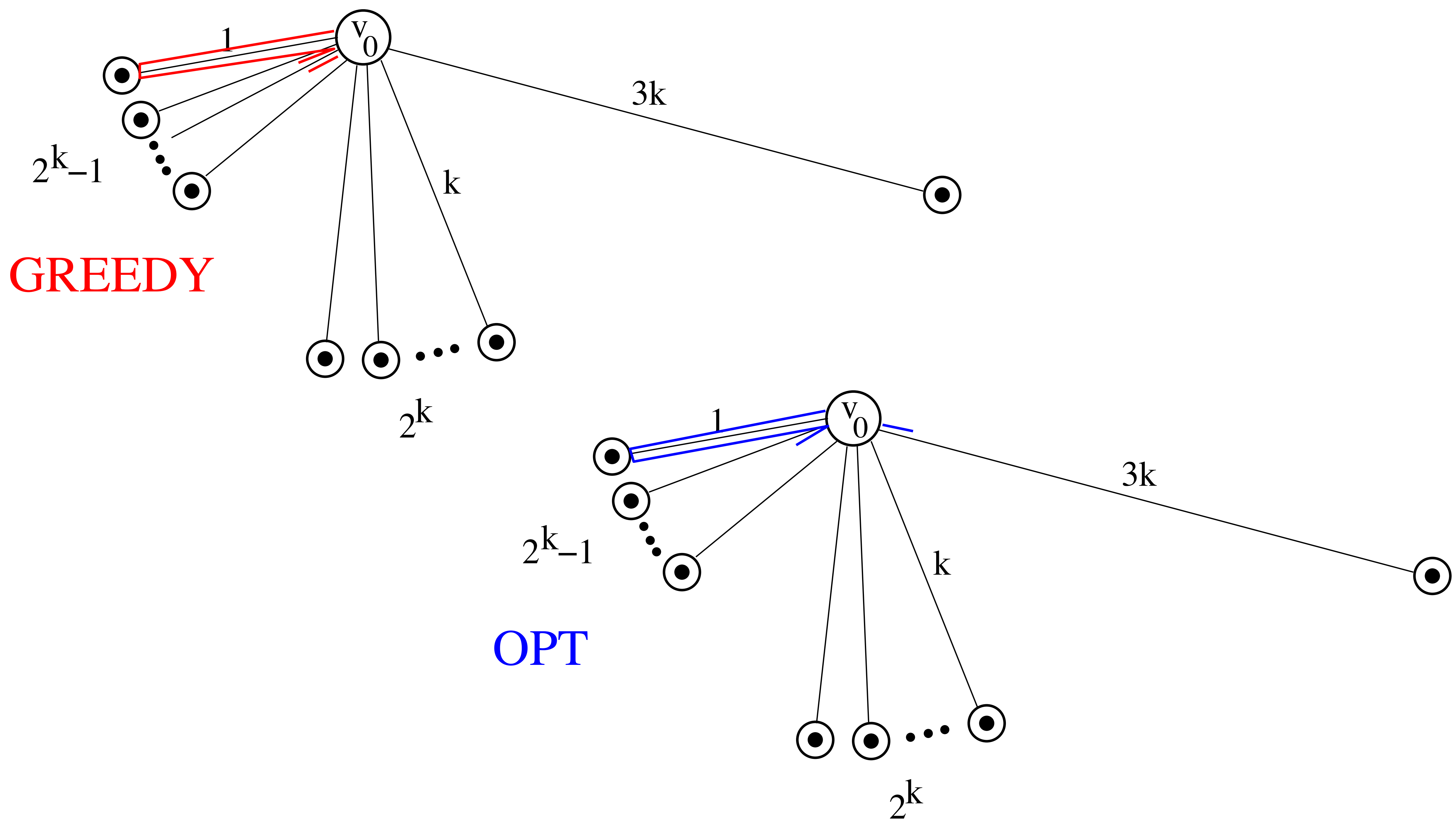
Amazingly, greedy is a 7/3-approx, and this bound is *tight*

Claim: Greedy is at best a 7/3-approx

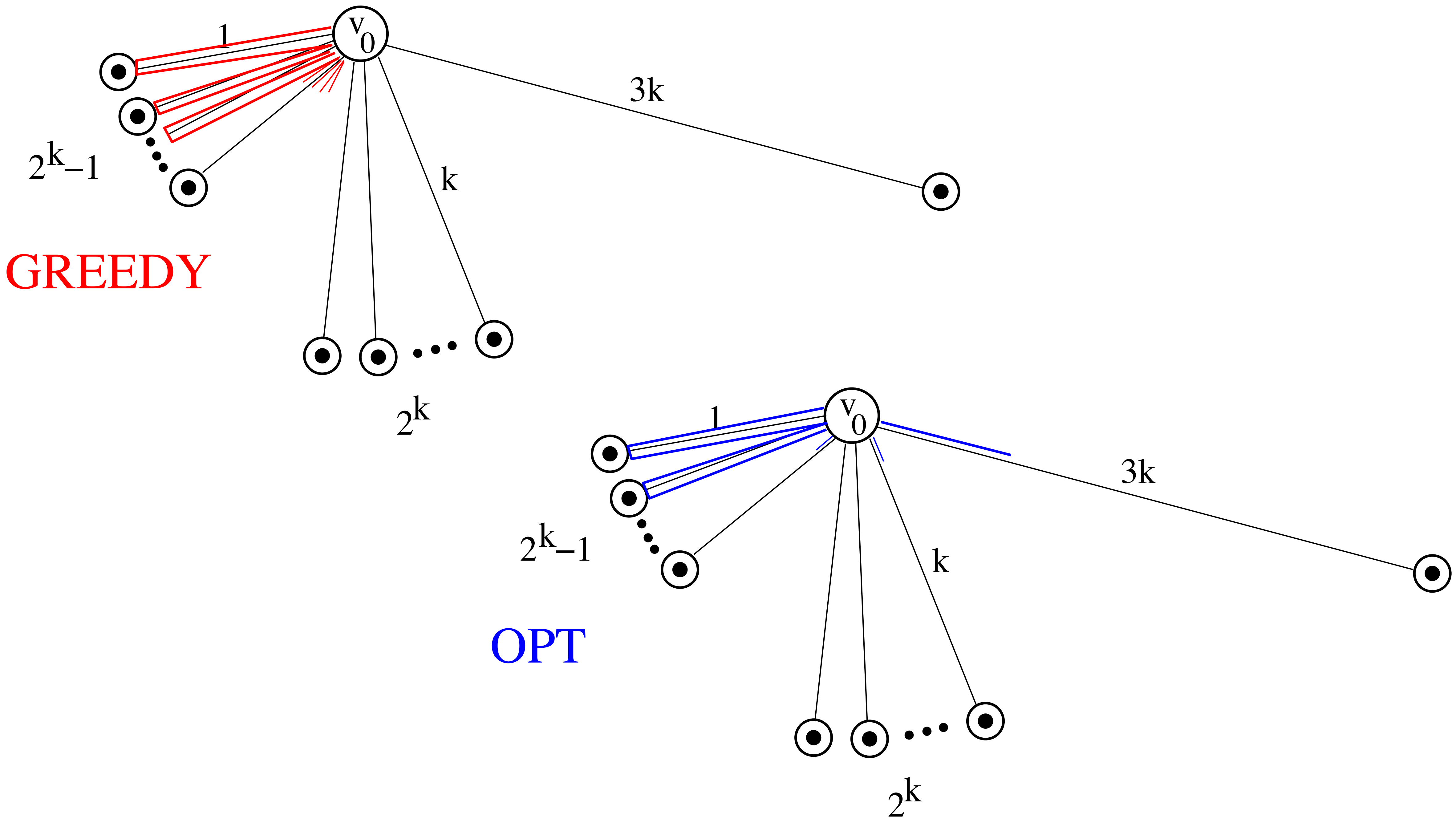
Example:



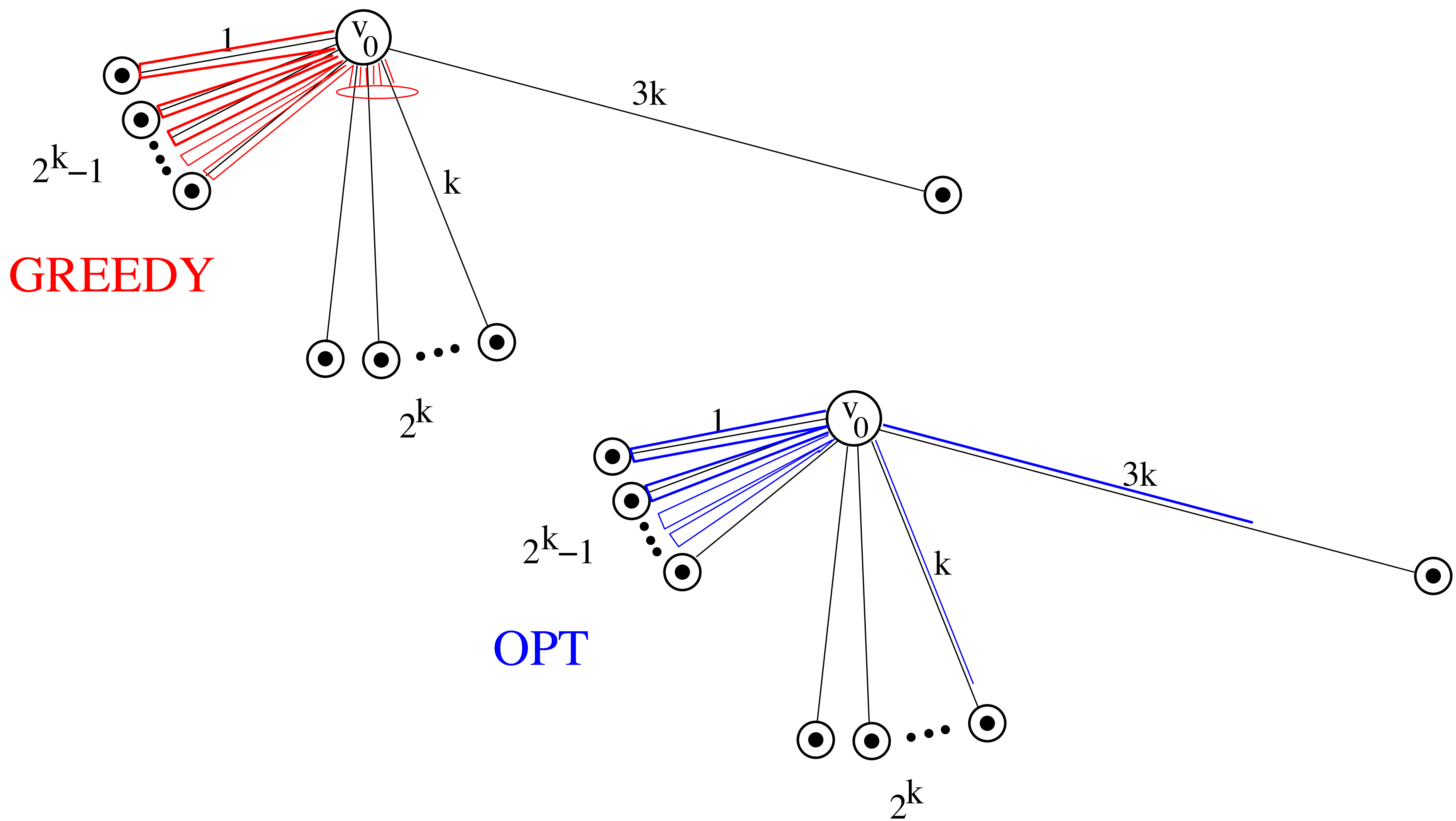
Time = 2:



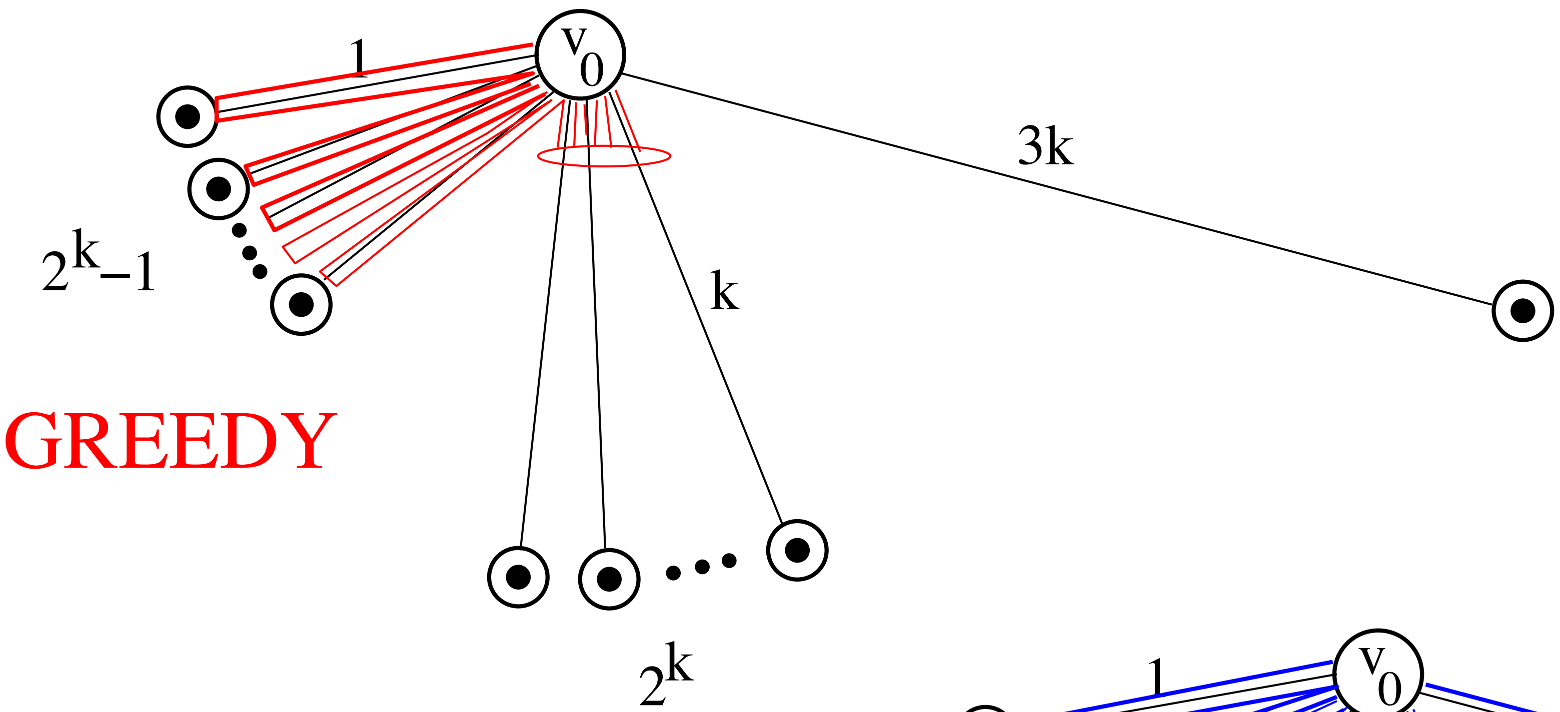
Time = 4:



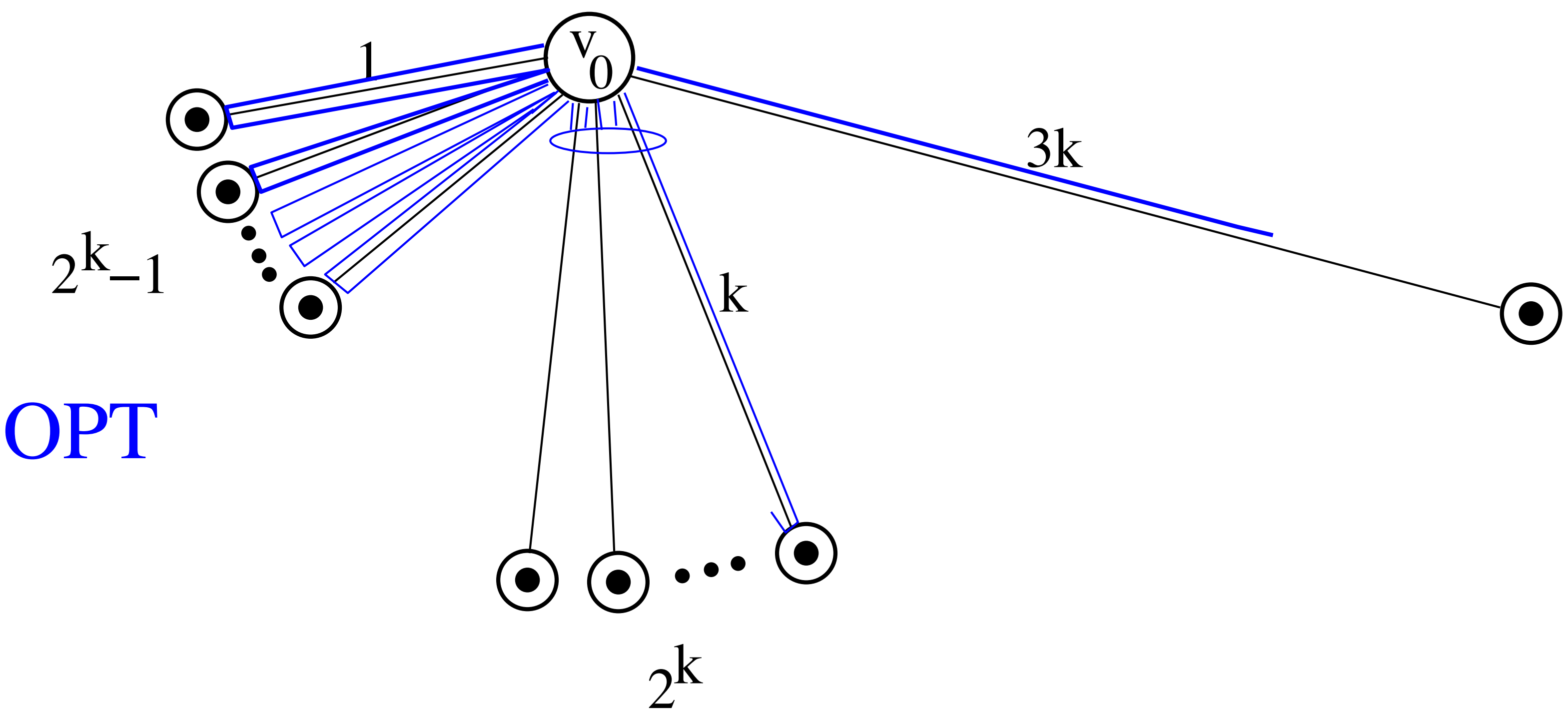
Time = 2k:



Time = 2k+4:

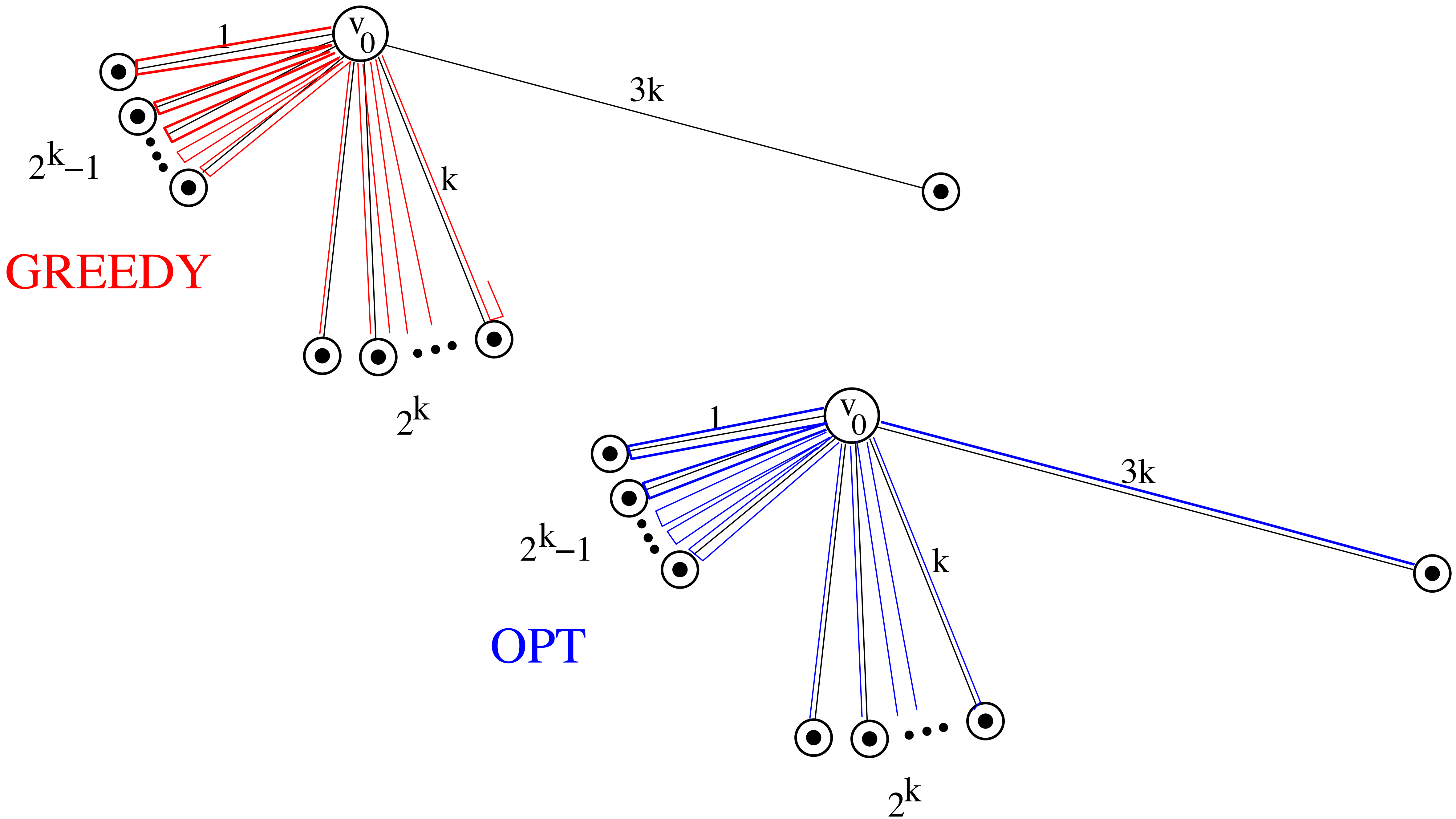


GREEDY



OPT

Time = 3k+2:



Greedy still has another $\approx 4k$ to go!

Analysis of Greedy on Stars:

Our analysis relies on the following fact, of independent interest:

Theorem: Greedy minimizes the *average completion time*

(completion time of robot i is the time when the robot is awake and no longer busy (moving))

Proof Idea: Exchange arguments

Analysis of Greedy on Stars:

Theorem: Greedy gives 7/3-approx in stars, and this is tight

(assume q asleep robots at each leaf initially)

NP-hardness for Stars:

Theorem: FTP is strongly NP-hard, even for weighted stars with $q = 1$ robot at each leaf

Proof: Reduction from NUMERICAL 3-DIM MATCHING (N3DM)
Input: Sets W, X, Y, each with n elements of integral sizes a_i, b_j, c_k, and a number d
Question: Can $W \cup X \cup Y$ be partitioned into n disjoint sets $S_1, \ldots, S_n$, with each S_h a triple (one element of W, of X, of Y) of size d?

WLOG: $n = 2^K$ $\quad a_i, b_j, c_k \leq d$

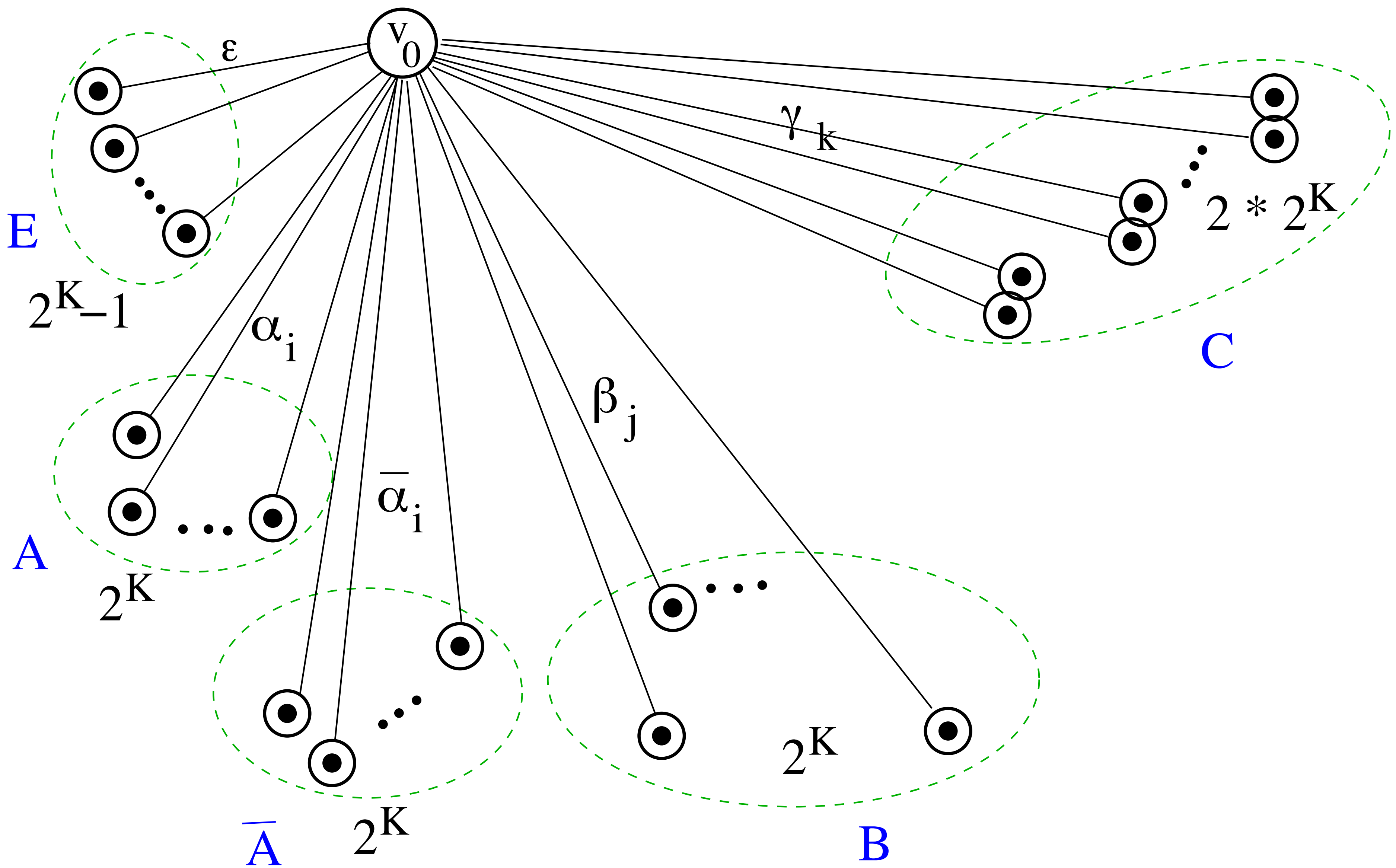
Let ε be sufficiently small $\quad (\varepsilon < 1/(2K))$

Let L be sufficiently large $\quad (L := 15d)$

Hardness Construction:

$\alpha_i := a_i/2 - \varepsilon K + d \qquad \overline{\alpha_i} := L - a_i - 2d$

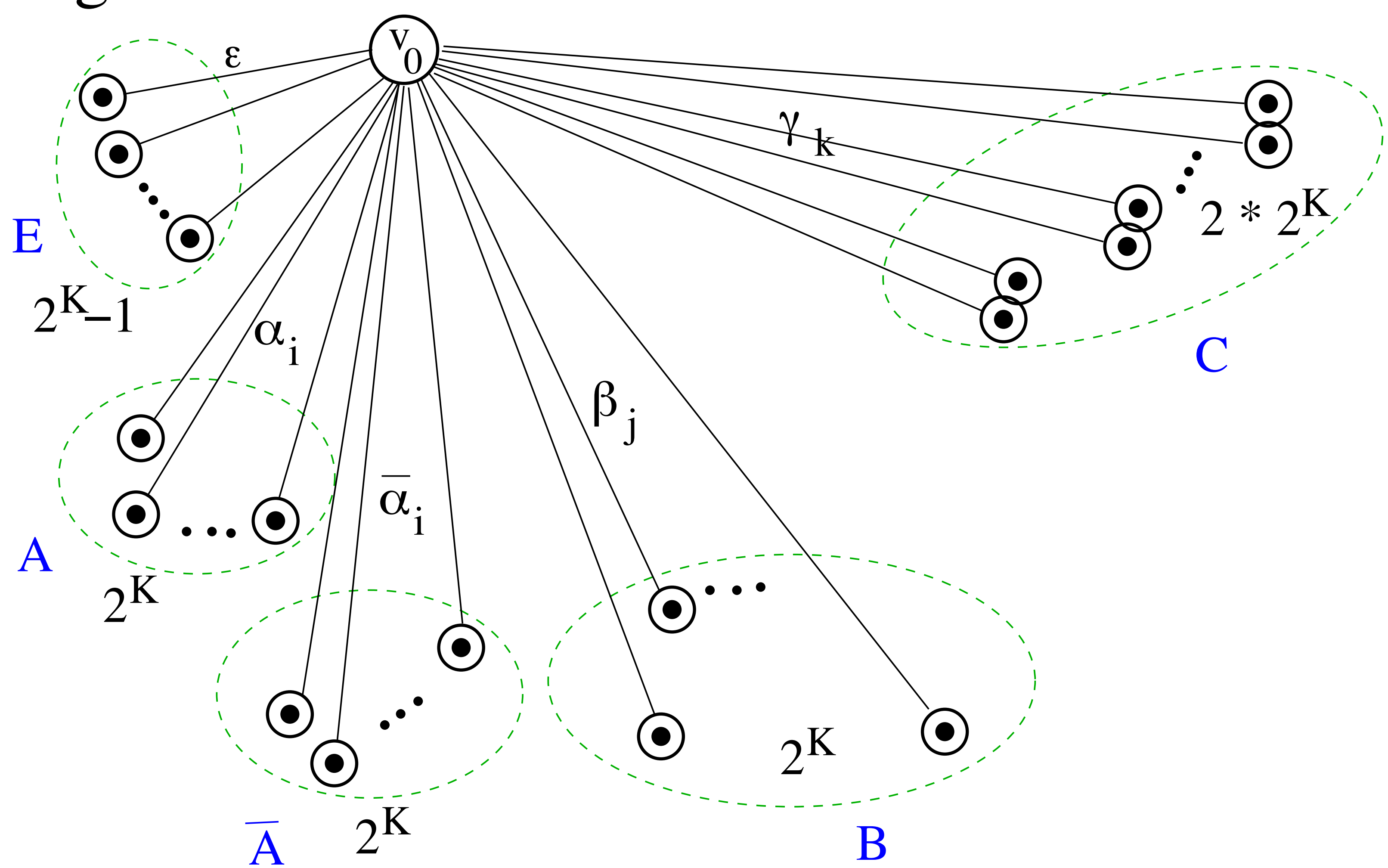
$\beta_j := b_j/2 + 2d \qquad \gamma_k := L - 7d + c_k$



Claim: $\exists$ schedule of makespan to awaken all robots within time L iff $\exists$ solution to N3DM

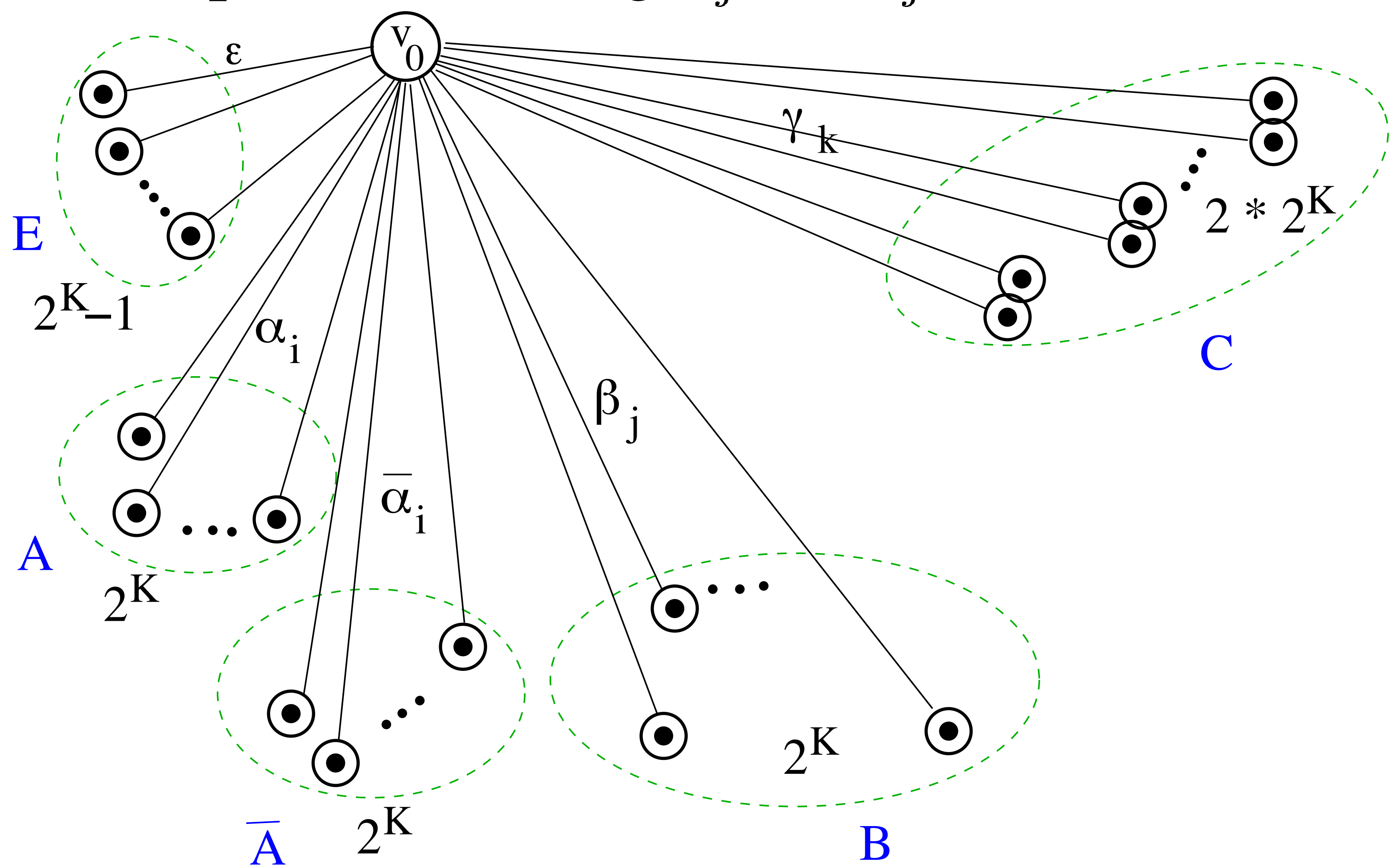
"IF": (**"ONLY IF"** is more involved)

- "greedy cascade" brings all $n = 2^K$ robots to v_0, time $2\epsilon K$
- these robots go to A-leaves



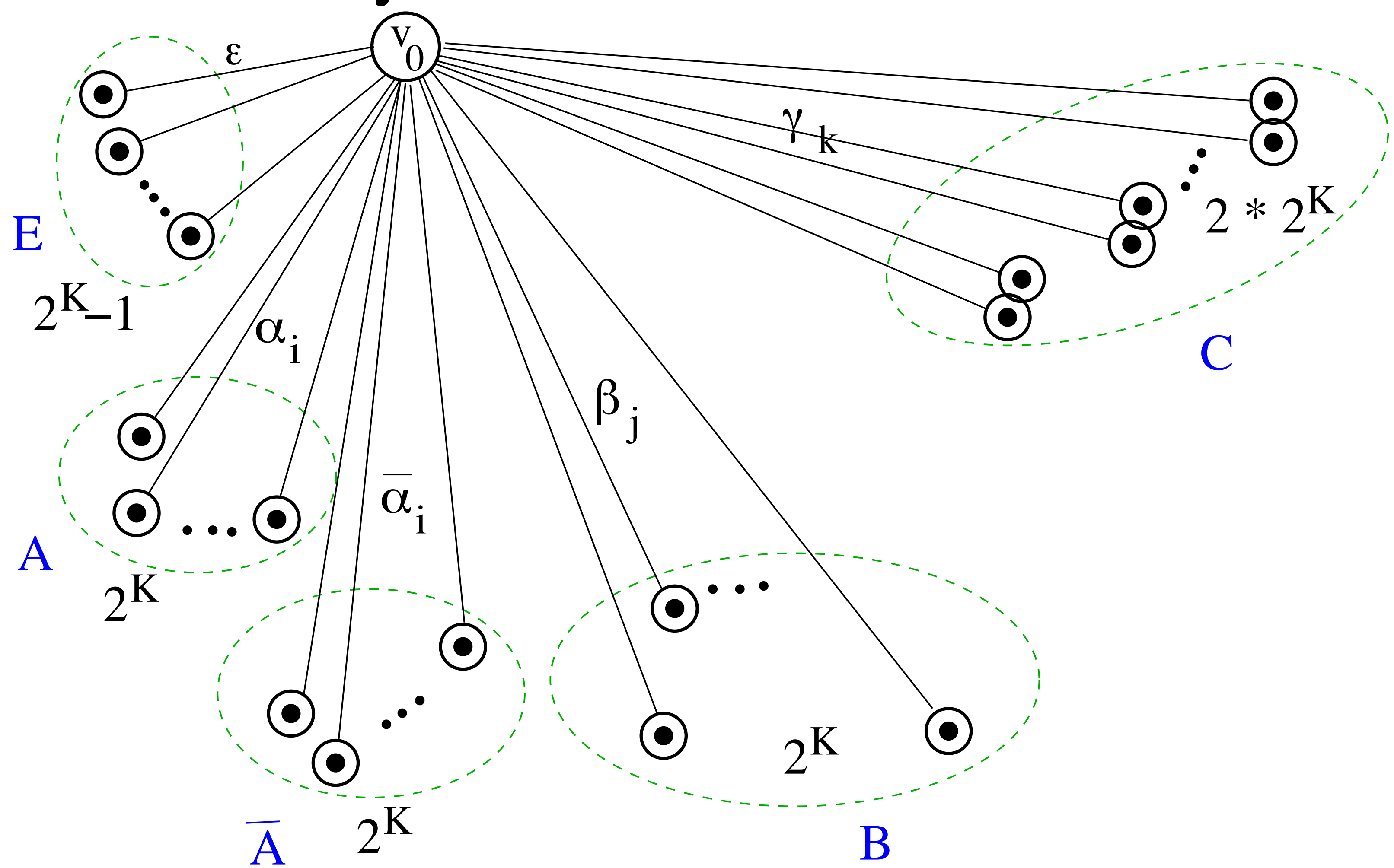
$\alpha_i := a_i/2 - \varepsilon K + d, \qquad \overline{\alpha_i} := L - a_i - 2d$

- 2 each return to v_0 at times $a_i + 2d$
- of each pair to $\overline{A}$, along α_i
- the other of each pair to B, along β_j: a_i, b_j in same S_h



$$\alpha_i := a_i/2 - \varepsilon K + d, \qquad \overline{\alpha_i} := L - a_i - 2d$$
$$\beta_j := b_j/2 + 2d, \qquad \gamma_k := L - 7d + c_k$$

- 2 robots get back to v_0 by time $a_i + b_j + 6d$
- they go to C, down pair of edges of length γ_k: $a_i + b_j + c_k = d$
- all robots at C awake by time L



$$\alpha_i := a_i/2 - \varepsilon K + d, \qquad \overline{\alpha_i} := L - a_i - 2d$$

$$\beta_j := b_j/2 + 2d, \qquad \gamma_k := L - 7d + c_k$$

PTAS for Stars with q Robots per Leaf:

Theorem: There is a PTAS for weighted stars with q robots at each leaf

Proof idea:

- Let $T \leq t^*$ be a good lower bound on makespan
  (use 3/7 times greedy solution)
- Partition edges: "short" (length $\leq \epsilon T$) and "long" (length $> \epsilon T$)
- Round up lengths of long edges to multiples of $\epsilon^2 T$
- Suffices to consider schedules in which each long edge is entered by a robot at a time that is a multiple of $\epsilon^2 T$
- Only $O(1/\epsilon^2)$ different lengths/start-times of long edges
- Enumerate (in $O(n^{O(1/\epsilon^4)})$) all possibilities of how many long edges of a given length are started at a given time
- In this way, we have "guesssed" the correct positions of all long edges
- Fill in short edges with a variant of greedy

General Stars with n_i Robots at Leaf v_i:

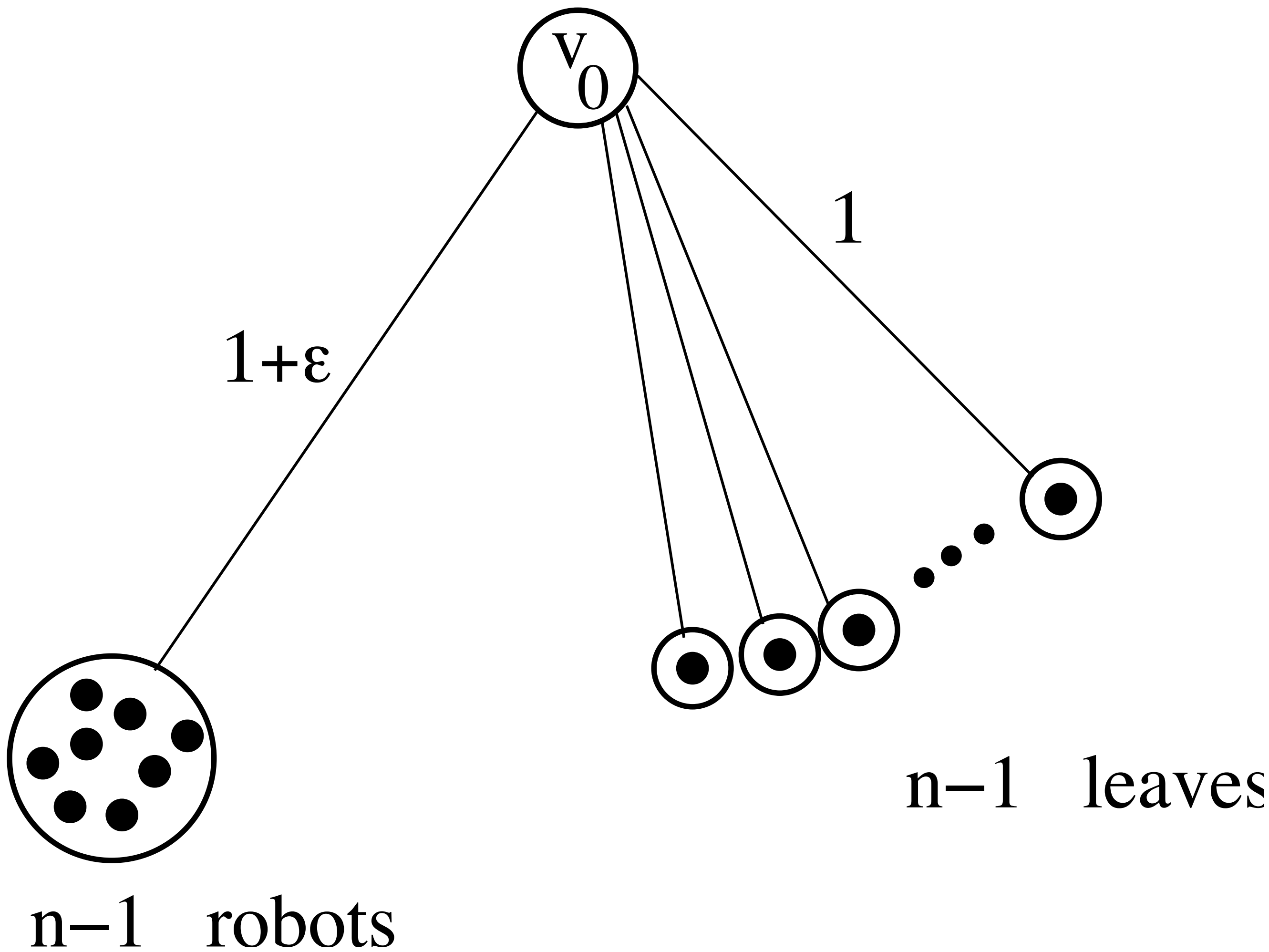
Now: *centroid metric*

(star with various spoke lengths, various n_i)

Consider a natural greedy strategy:

"Shortest Edge First" (SEF): An awake robot at v_0 picks a shortest edge leading to asleep robots, breaking ties according to # robots

Example: Shortest Edge First Can be Bad:



SEF: $\Theta(\log n)$ **OPT:** $O(1)$

Key Dilemma: Choose a short edge leading to few robots or a long edge leading to many?

Devising an Alternative Strategy:

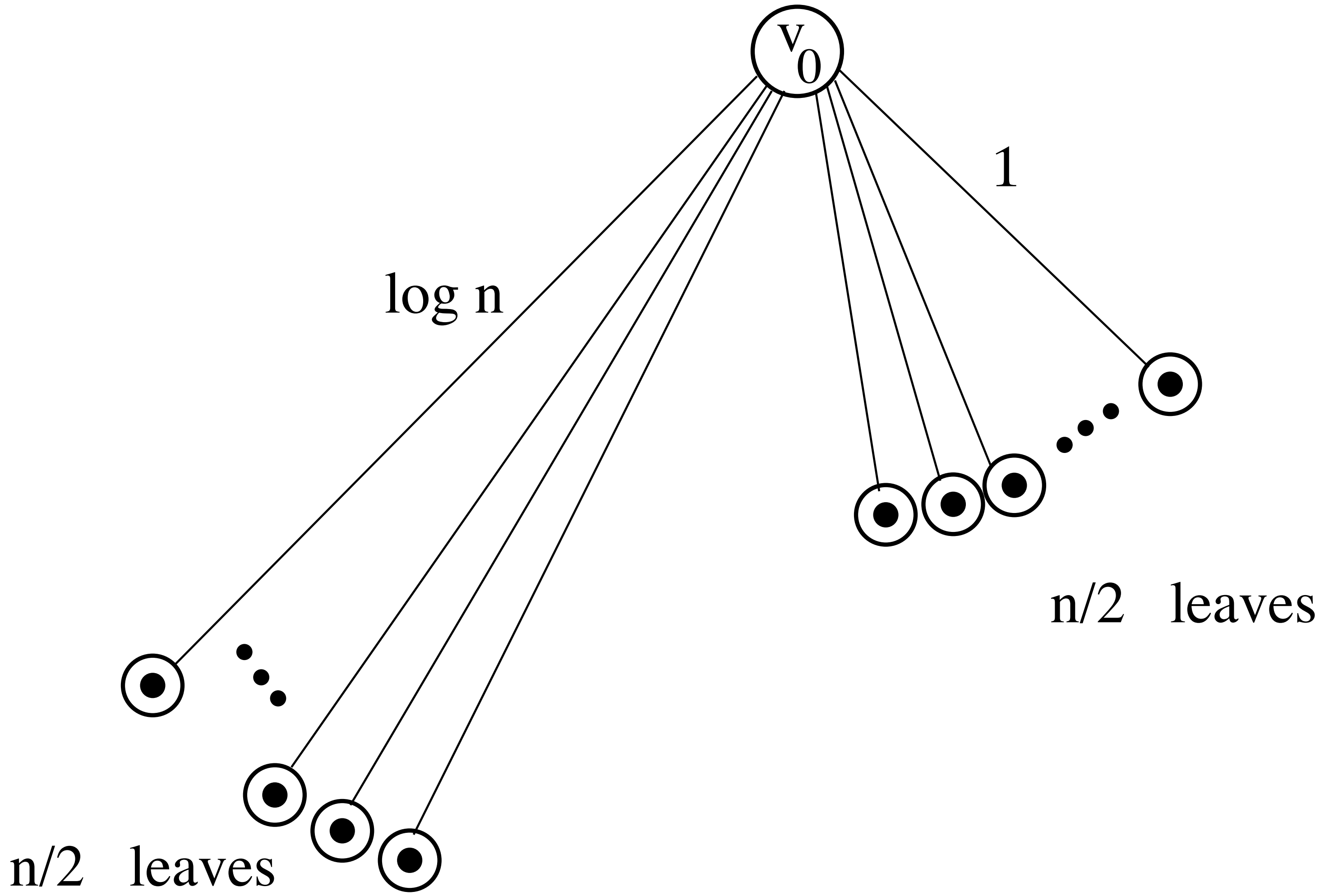
Round edge lengths to powers of 2 (may double approx factor)

Issue: How to pick what length class to visit first?

Idea: Hedge our bets by repeated doubling

"Repeated Doubling" (RD): Edge lengths traversed repeatedly (roughly) double in size; in each length class be greedy in # robots

Example: Repeated Doubling Can be Bad:



RD: $\Theta(\log^2 n)$ **OPT:** $O(\log n)$

Idea: Merge the SEF and RD strategies

Tag-Team Algorithm:

Awaken one edge in each length class $1, 2, 4, 8, \ldots$, **but** before going to the next length class, awaken the shortest available edge.

This *combination* of two $\Theta(\log n)$-approx methods yields an $O(1)$-approx!

Theorem: The Tag-Team Algorithm gives a 14-approx

General Weighted Graphs:

General graph $G = (V, E)$ with non-negative edge weights

$r(v)$ =# asleep robots at v $\qquad$ $\delta(v)$ =degree of v

Lemma: Suppose $r(v_0) \geq \delta(v_0)$ for the source node v_0, and $r(v_i) \geq \delta(v_i) - 1$ at any other node in G. Then the FTP can be solved by breadth-first search.

General Weighted Graphs:

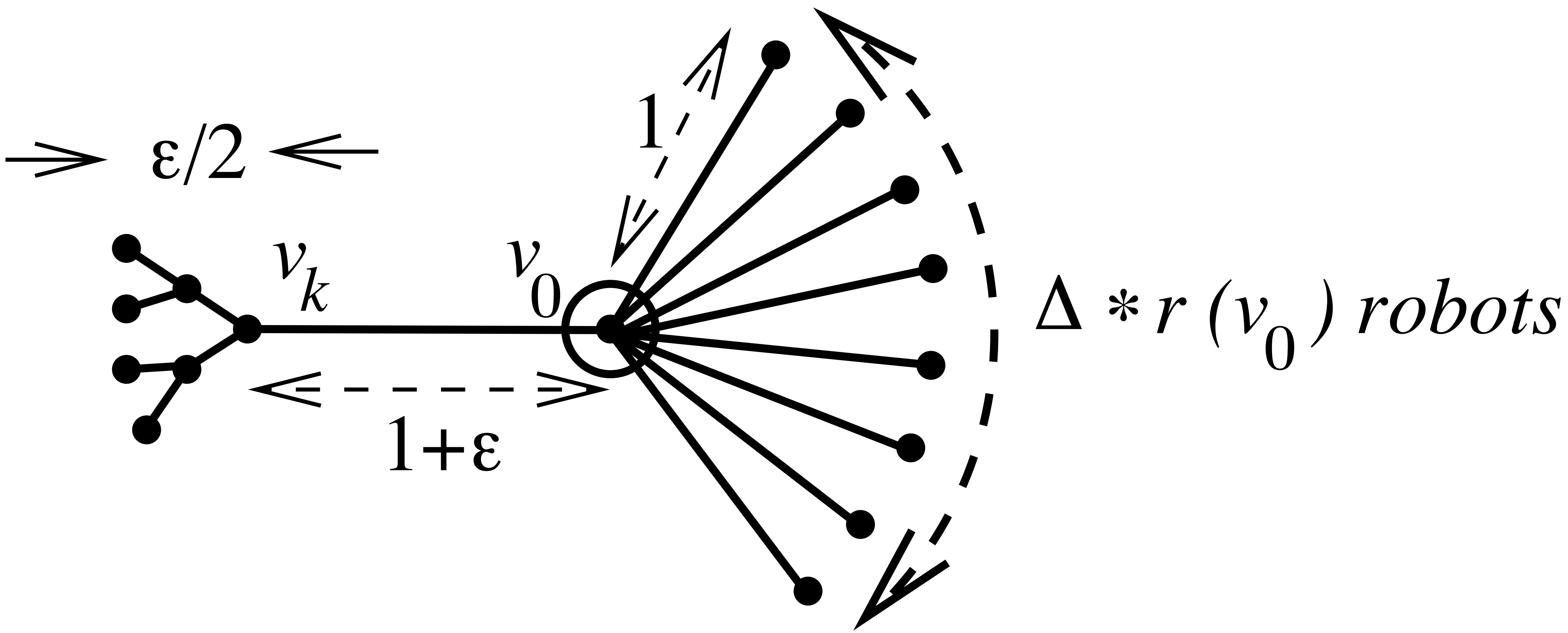
$$\Delta_G := \max\{\frac{\delta(v_0)}{r(v_0)}, \frac{\delta(v_i) - 1}{r(v_i)}, i = 1, \ldots, n-1\}$$

Theorem: There is a linear-time on-line algorithm for the FTP in G that guarantees a competitive ratio of $O(\log \Delta_G)$.

Proof Sketch: Simulate BFS: At v_i use a greedy strategy to wake up all robots adjacent to v_i, with binary tree of depth $\lceil \log \frac{\delta(v_0)}{r(v_0)} \rceil$ for the root, and $\lceil \log \frac{\delta(v_i)-1}{r(v_i)} \rceil$ for v_i

Greedy implies that any edge e in the resulting wake-up tree can only be placed below edges f that satisfy $w_f \leq w_e$.

Bad Example for Local Greedy Strategy:



Local greedy takes $\Theta(\log n)$; OPT takes $3(1+\epsilon)$

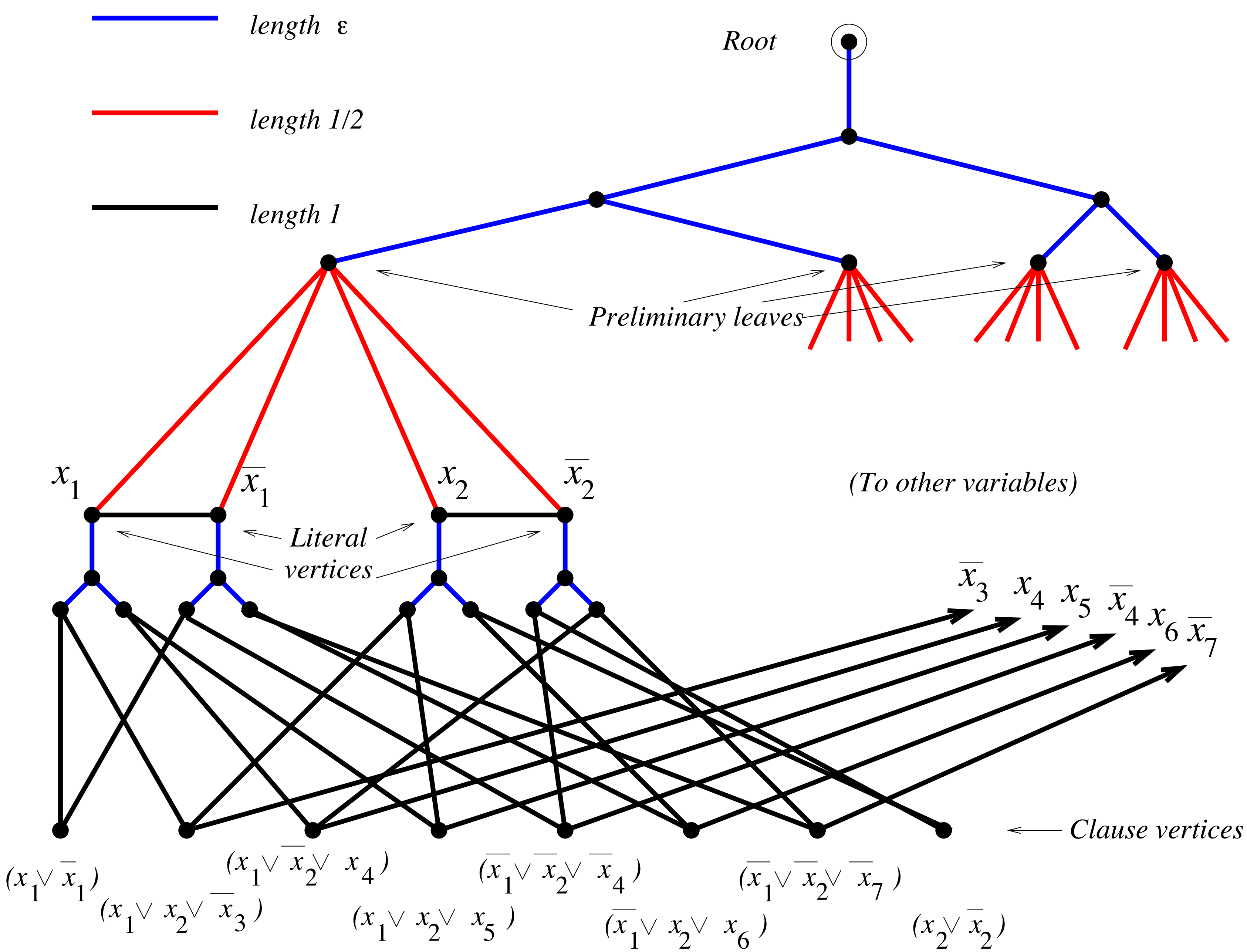
Hardness of Approx:

- FTP is NP-hard even if *one* high-degree vertex (v_0)

Question: If degrees are all small, can we do much better?

Theorem: Even if all nodes have degree ≤ 5 and at most one robot per node, it is NP-hard to get better than 5/3-approx.

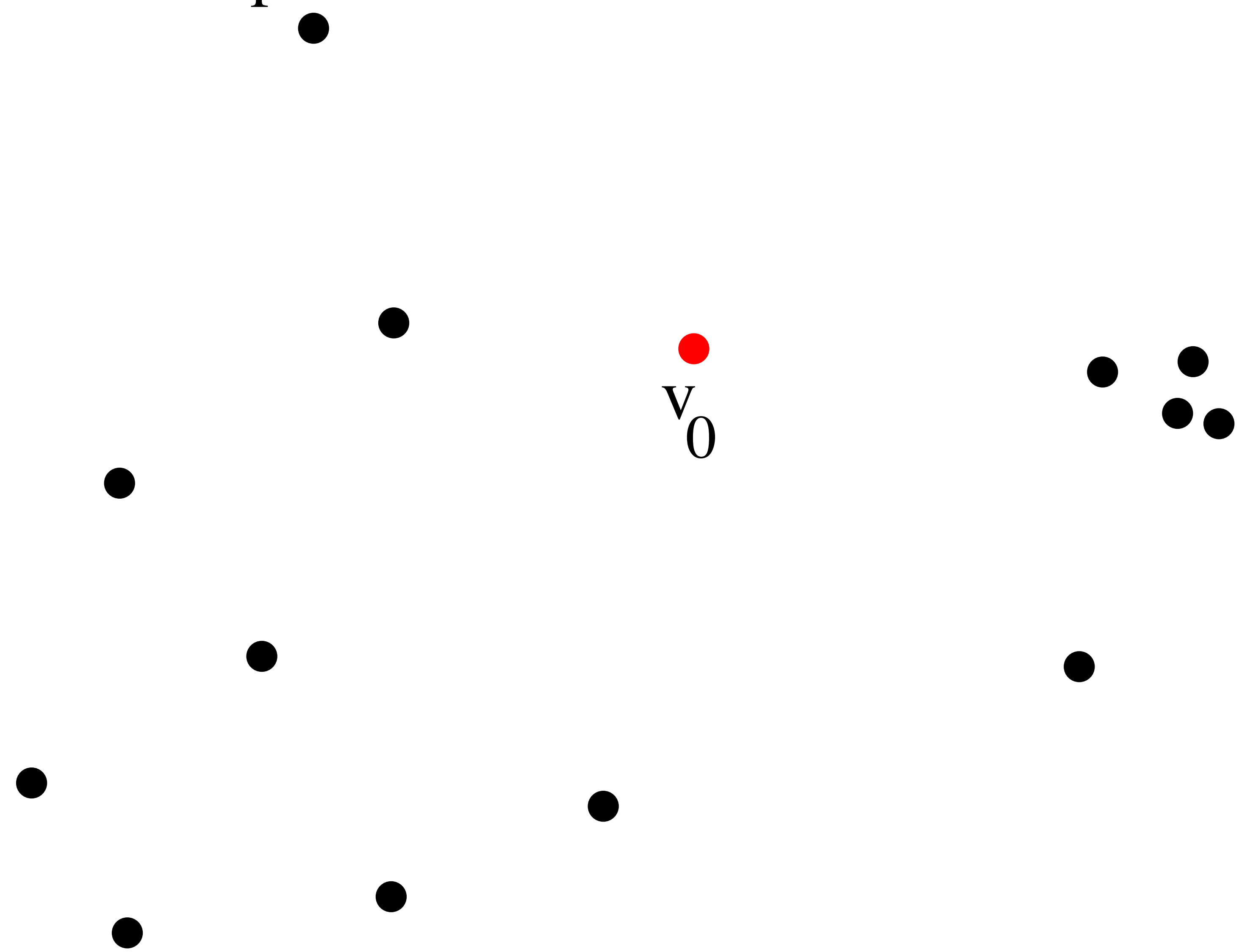
Proof Sketch: From 3SAT:



A solution with makespan $3/2+O(\epsilon \log n)$ if $\exists$ satisfying truth assignment; makespan $\geq 5/2$ if none (pick $\epsilon = o(\log n)$)

Geometric Instances:

Robots at points in the plane:



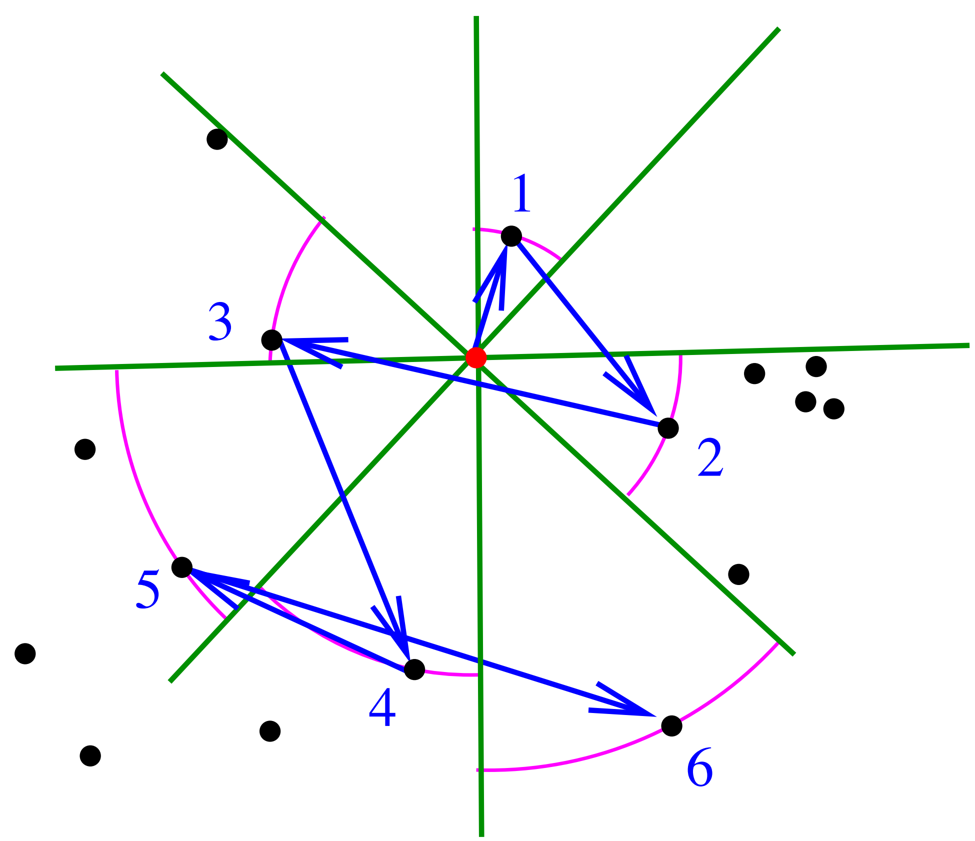
Question: Can we exploit geometry to get good approx?

OPEN: Is the problem NP-hard?

Geometric Instances: $O(1)$-approx:

Theorem: $\exists$ an $O(1)$-approx, time $O(n \log n)$, for the geometric FTP in any fixed dimension d. The algorithm gives wake-up schedule with makespan $O(diam(R))$.

Strategy: When robot at p awakens, it awakens nearest asleep robot in each of K sectors, in order of increasing distance from p



- $G_K = (R, E_K)$ is a Θ-graph (which is a t-spanner)

- Distances in G_K approx Euclidean

- Let v_ℓ be the last robot to be awakened

- If robot at point v is awakened at time t, then all neighbors of v in G_K are awakened by time $t + \xi$, where $\xi =$ length of path $v, u_1, u_2, \ldots, u_j$.

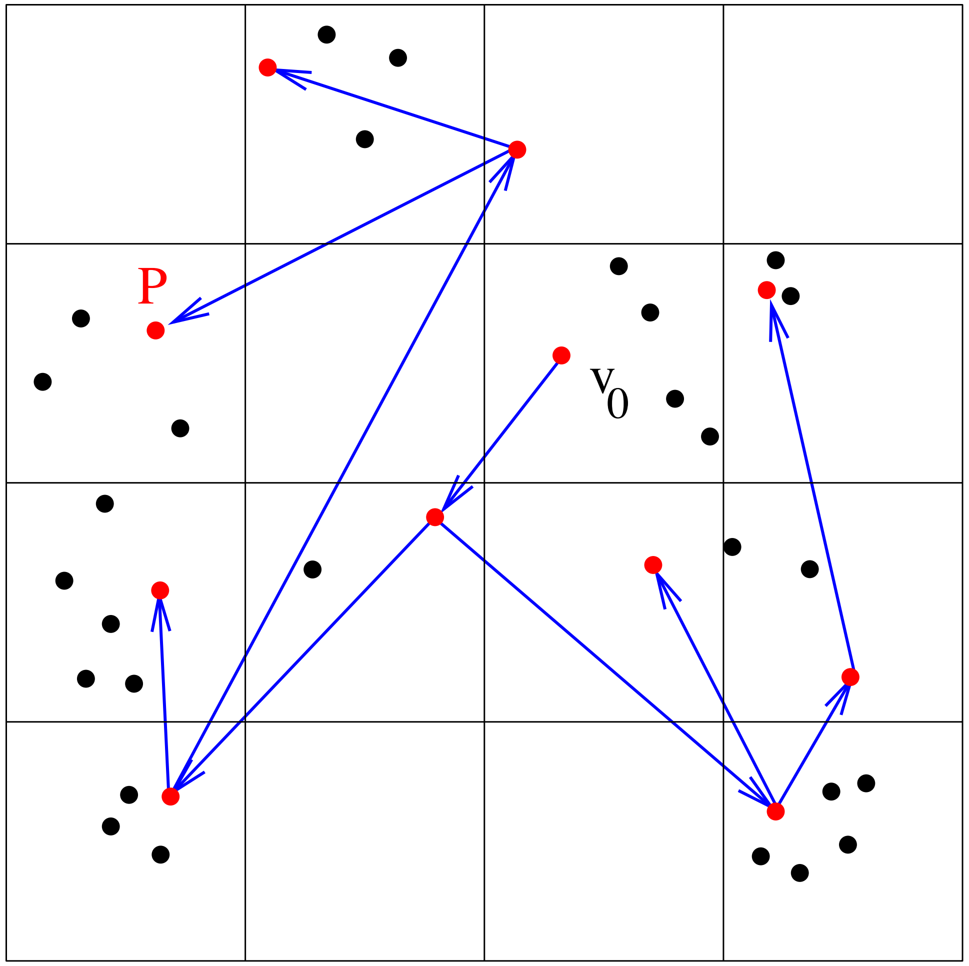
- $\xi \leq (2j-1) \cdot d(v, u_j) \leq (2K-1) \cdot d(v, u_j)$

- The path from v_0 to v_ℓ in the wake-up tree has length at most $(2K-1) \cdot d_{G_K}(v_0, v_\ell) \leq O(1) \cdot d(v_0, v_\ell)$

PTAS for Geometric Instances:

Rescale so that all robots lie in unit square

Look at m-by-m grid of pixels, $m = O(1/\epsilon)$



Consider an enumeration over a special class of wake-up trees on a set P of representative points, one per occuppied pixel

A wake-up tree is *pseudo-balanced* if each root-to-leaf path has $O(\log^2 n)$ nodes

ALGORITHM:

0. Pick a representative point in each occuppied pixel $\rightarrow$ set P

1. Among all pseudo-balanced wake-up trees for P, pick one ($\mathcal{T}_b^*(P)$) of min makespan, $t_b^*(P)$

   (outdegree at most $\min\{m^2 - 1, \ell + 1\}$ if ℓ robots in pixel)

Only $2^{O(m^2 \log m)}$ trees.

2. Convert $\mathcal{T}_b^*(P)$ into a wake-up tree for *all* robots by replacing each $p \in P$ with an $O(1)$-approx wake-up tree for robots in p's pixel
(time $O(n \log n)$)

Total time: $O(2^{O(m^2 \log m)} + n \log n)$

Correctness:

Lemma 1. There is a choice of representative points P such that the makespan of an optimal wake-up tree of P is at most $t^*(R)$.

Proof: Just pick the representative point to be the location of the first robot that is awakened in an opt solution, $\mathcal{T}^*(R)$, for the set R of all robots.

Lemma 2. If $\exists$ wake-up tree, $\mathcal{T}$, of makespan t, then, for any $\mu > 0$, there exists a pseudo-balanced awakening tree, $\mathcal{T}_b$, of makespan $t_b \le (1+\mu)t$.

Proof Sketch:

Use a heavy path decomposition of $\mathcal{T}$

Any root-to-leaf path has only $O(\log n)$ light edges

Decompose each heavy path into short subpaths of length $\xi = \mu t / \log n$

(only $O((1 + 1/\mu) \log n)$ subpaths on any root-to-leaf path of $\mathcal{T}$)

Modify wake-up tree $\mathcal{T}$ to transform each subpath into a wake-up tree of height $O(\log n)$, with a small increase in makespan

Lemma 3. For any two choices, P and P', of the set of representative points, $t_b^*(P) \leq t_b^*(P') + O((\log^2 m)/m)$.

Proof: Pixels have size $O(1/m)$ and there are at most $O(\log^2 m)$ awakenings in each root-to-leaf path of a pseudo-balanced tree; thus, any additional wake-up cost is bounded by $O((\log^2 m)/m)$.

Lemma 4. For any pseudo-balanced wake-up tree of P, there exists a wake-up tree, $\mathcal{T}(R)$, with makespan $t(R) \leq t_b(P) + O((\log^2 m)/m)$.

Putting the Pieces Together:

Theorem: There is a PTAS, with running time $O(2^{O(m^2 \log m)} + n \log n)$, for the geometric FTP in any fixed dimension d.

Proof:

The makespan, t, of the wake-up tree we compute obeys:

$$\begin{aligned} t &\leq t_b^*(P) + O((\log^2 m)/m) \\ &\leq t_b^*(P') + 2 \cdot O((\log^2 m)/m) \\ &\leq (1+\mu)t^* + O((\log^2 m)/m) \\ &\leq t^* \left(1 + \mu + \frac{C \log^2 m}{m}\right) \\ &\leq t^*(1+\epsilon), \end{aligned}$$

for appropriate choices of μ and m, depending on ϵ.

Experimental Studies [with Marcelo Sztainberg]:

Experimental analysis of 3 natural heuristics, including greedy

Analysis of greedy on geometric data: $\Omega(\sqrt{\log n})$-approx

Conclusion – Summary of Results:

1. FTP is NP-hard, even for stars, with one robot per leaf
2. $O(1)$-approx for (general) stars
3. PTAS for stars, same number of robots at each leaf
4. Tight analysis of greedy heuristic on stars: 7/3-approx
5. $o(\log n)$-approx for FTP in ultrametrics ($2^{O(\sqrt{\log\log n})}$-approx)
6. Simple linear-time on-line algorithm, $O(\log \Delta)$-competitive
   ($\Delta \leq$ max degree)
7. NP-hard to get <5/3-approx in offline problem, even if $\Delta \leq 5$
8. PTAS for geometric instances in fixed-dimension, L_p metric
   Time $O(n \log n + 2^{poly(1/\varepsilon)})$

Conclusion – Open Problems:

OPEN: Is FTP in the Euclidean plane NP-hard?

OPEN: Is there an $o(\log n)$-approx for general metric spaces?

Angular Freeze Tag

Angular Freeze Tag

Beam It Up, Scotty: Angular Freeze-Tag with Directional Antennas*

Sándor P. Fekete[1] and Dominik Krupke[1]

1 TU Braunschweig
{s.fekete,d.krupke}@tu-bs.de

Abstract

We consider distributing mission data among the members of a satellite swarm. In this process, spacecraft cannot be reached all at once by a single broadcast, because transmission requires the use of highly focused directional antennas. As a consequence, a spacecraft can transmit data to another satellite only if its antenna is aiming right at the recipient; this may require adjusting the orientation of the transmitter, incurring a time cost proportional to the required angle of rotation. The task is to minimize the total distribution time. This makes the problem similar in nature to the *Freeze-Tag Problem* of waking up a set of sleeping robots, but with angular cost at vertices, instead of distance cost along the edges of a graph. We prove that approximating the minimum length of a schedule for this *Angular Free-Tag Problem* within a factor of less than 5/3 is NP-complete, and provide a 9-approximation for the 2-dimensional case that works even in online settings with incomplete information. Furthermore, we develop an exact method based on Mixed Integer Programming that works in arbitrary dimensions and can compute provably optimal solutions for benchmark instances with about a dozen satellites.

1 Introduction

Technische Universität Braunschweig

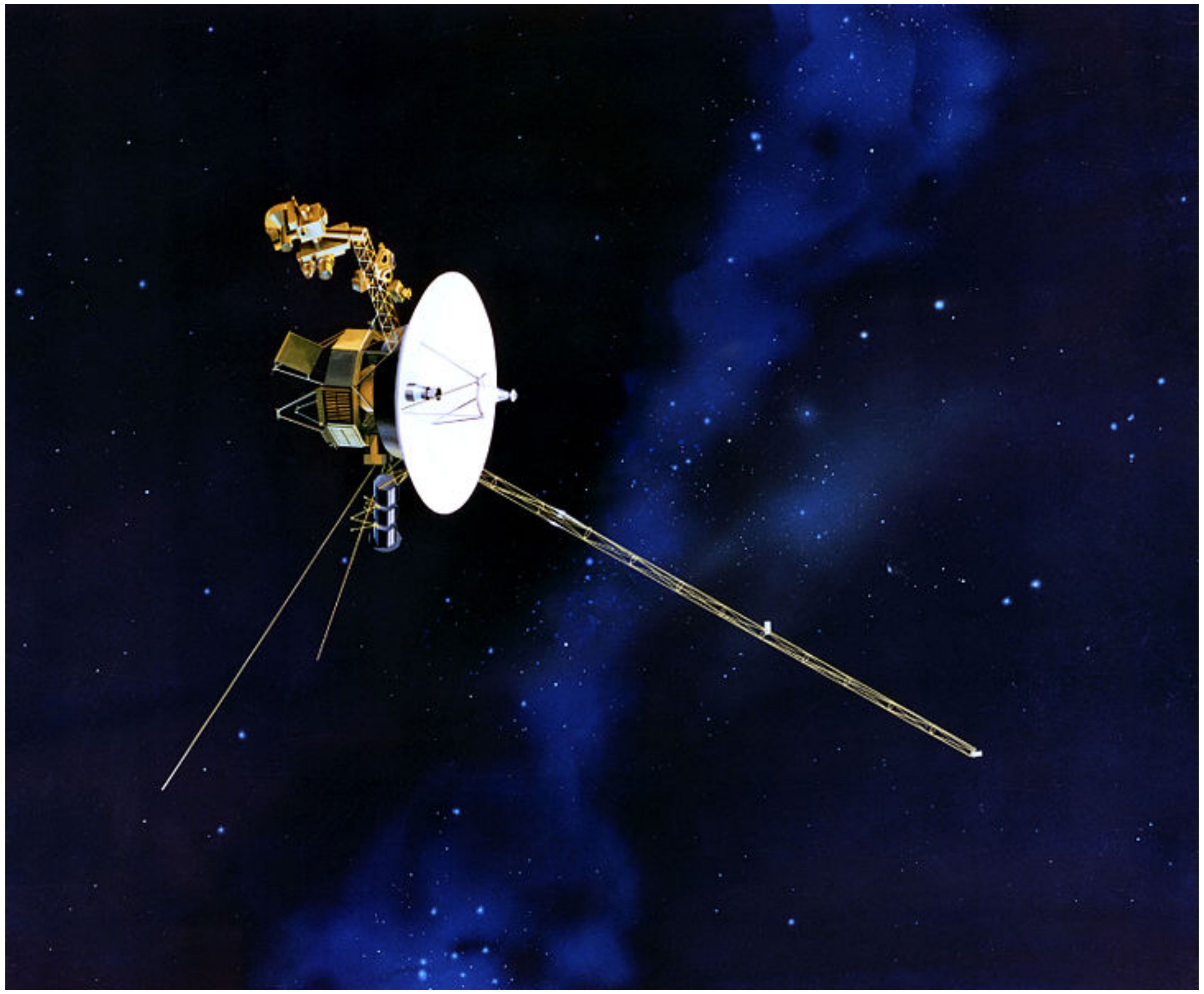
Institute of Operating Systems and Computer Networks

Beam It Up, Scotty: Angular Freeze-Tag with Directional Antennas

Sándor P. Fekete and Dominik Krupke

March 20, 2018

Motivation

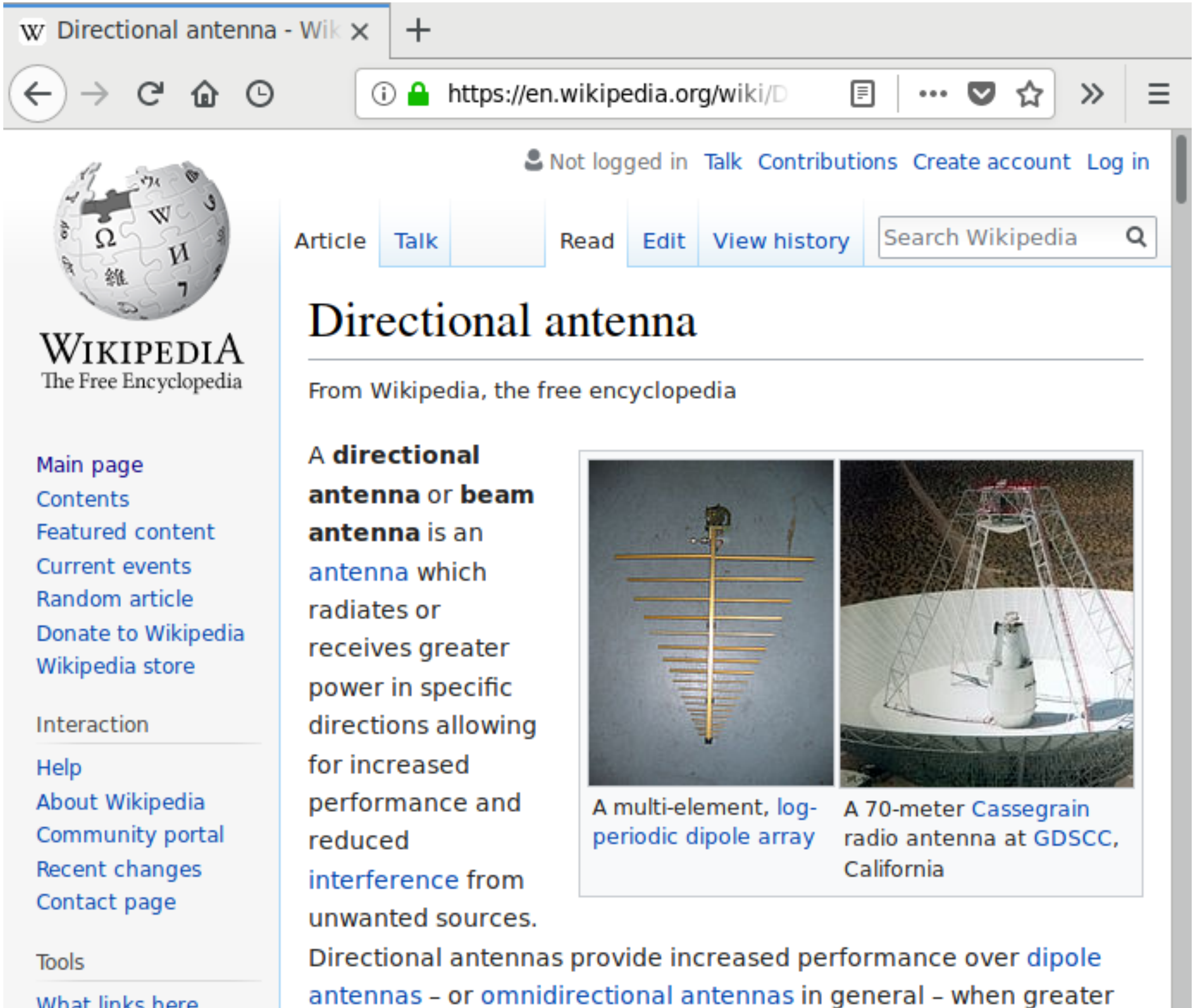


- Highly focused antennas.
- Expensive rotations.

How can we quickly distribute information from one satellite to all others?

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Wikipedia


Directional antenna - Wik
https://en.wikipedia.org/wiki/D
Not logged in Talk Contributions Create account Log in
WIKIPEDIA
The Free Encyclopedia
Article Talk
Read Edit View history
Search Wikipedia
Directional antenna
From Wikipedia, the free encyclopedia
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store
Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page
Tools
What links here
A directional antenna or beam antenna is an antenna which radiates or receives greater power in specific directions allowing for increased performance and reduced interference from unwanted sources.
A multi-element, log-periodic dipole array
A 70-meter Cassegrain radio antenna at GDSCC, California
Directional antennas provide increased performance over dipole antennas – or omnidirectional antennas in general – when greater

Wikipedia

Directional antenna - Wik ×

https://en.wikipedia.org/wiki/D

the antenna but for small antennas can be increased by adding a ferrite rod), and efficiency (again, affected by size, but also resistivity of the materials used and impedance matching). These factors are easy to improve without adjusting other features of the antennas or coincidentally improved by the same factors that increase directivity, and so are typically not emphasized.

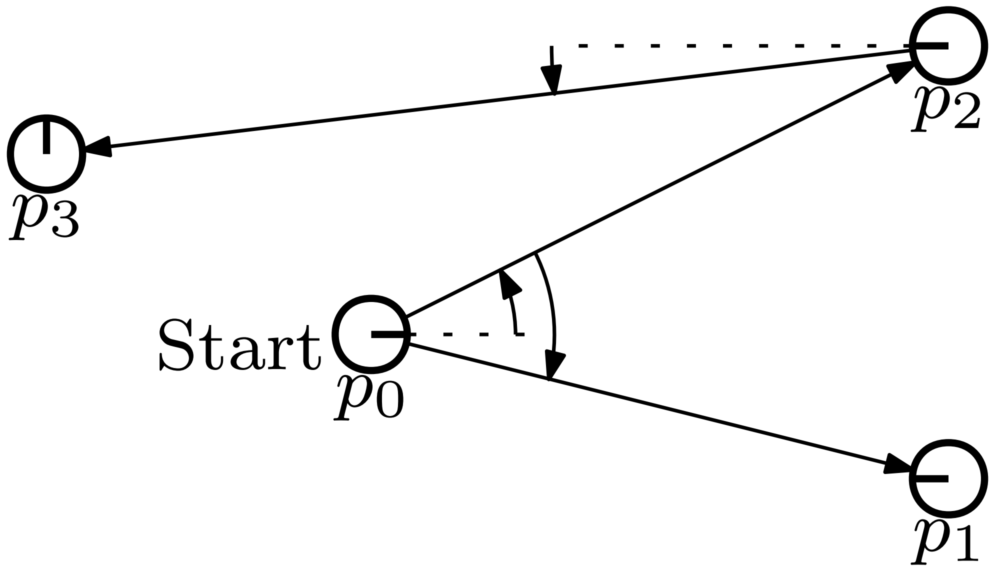
Applications [edit]

High gain antennas are typically the largest component of deep space probes, and the highest gain radio antennas are physically enormous structures, such as the Arecibo Observatory. The Deep Space Network uses 35 m dishes at about 1 cm wavelengths. This combination gives the antenna gain of about 100,000,000 (or 80 dB, as normally measured), making the transmitter appear about 100 million times stronger, and a receiver about 100 million times more sensitive, *provided the target is within the beam*. This beam can cover at most one hundred millionth (10^{-8}) of the sky, so very accurate pointing is required.

Gallery [edit]

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

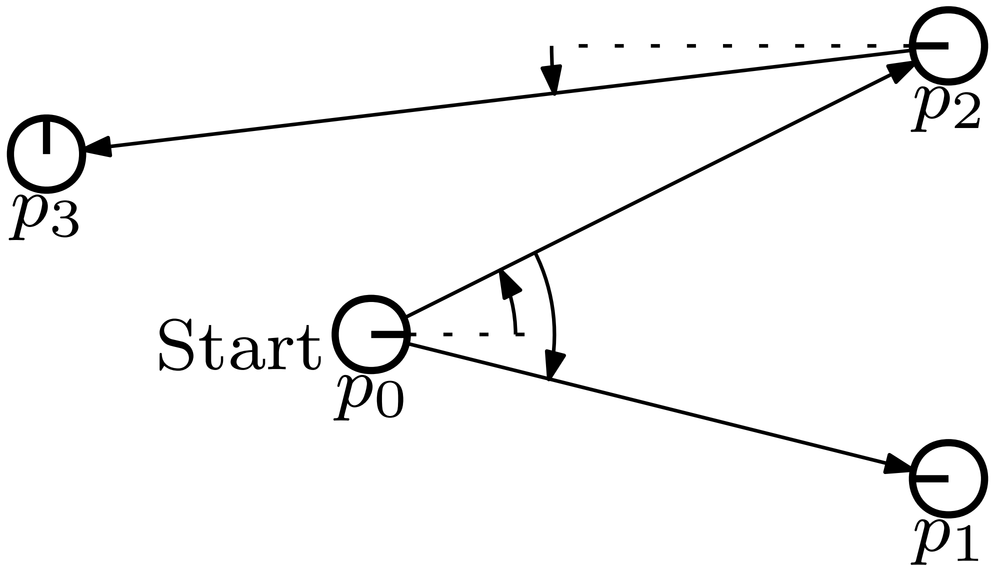


d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

Some simplifications:



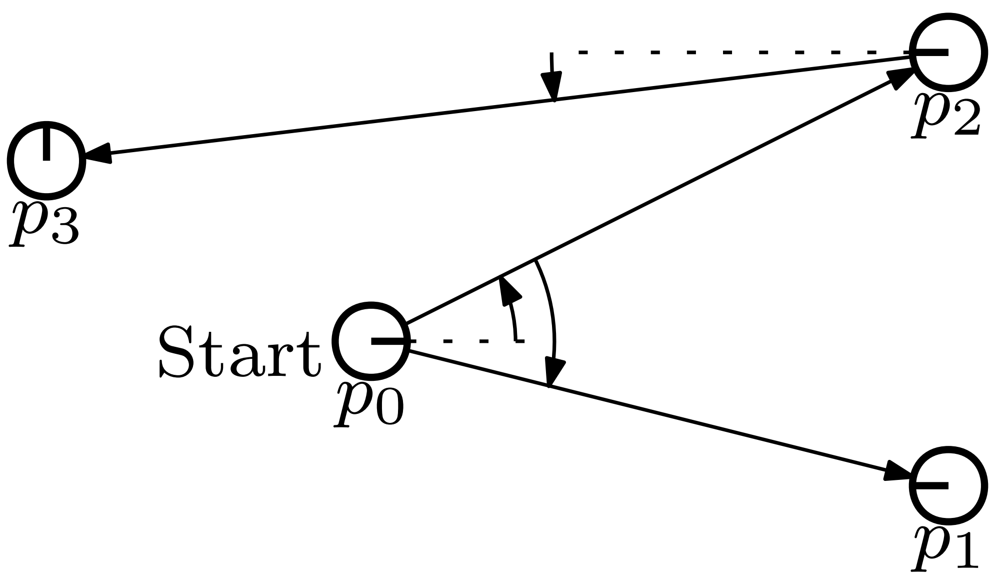
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

Some simplifications:

1. Only sender has to adjust.



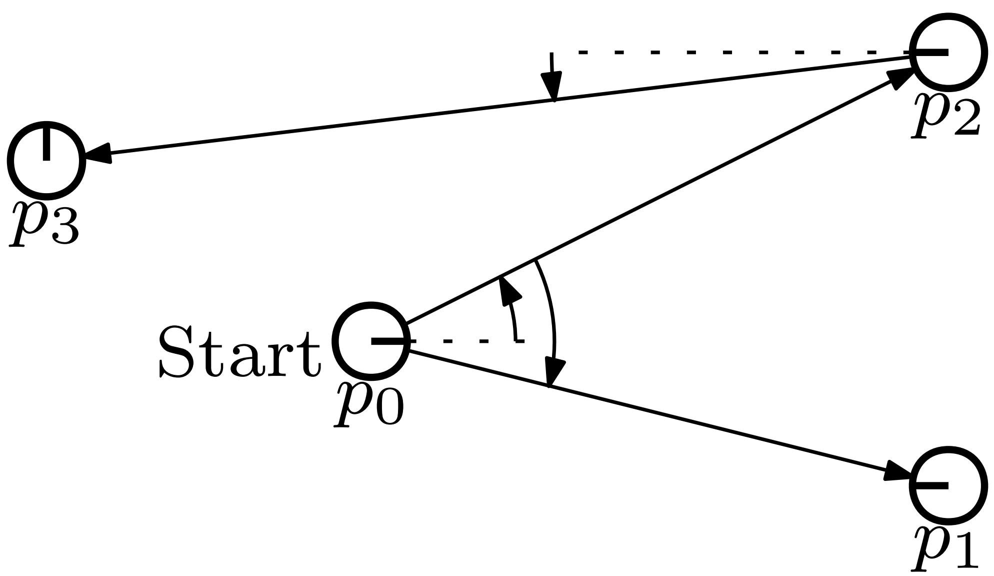
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

Some simplifications:

1. Only sender has to adjust.
2. Exact adjustment, i.e. beam is a ray.



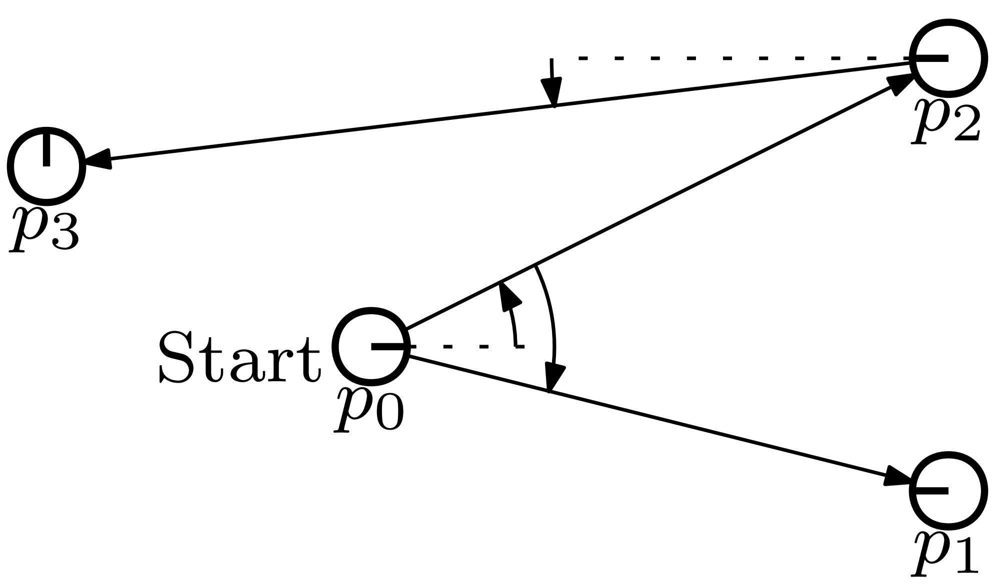
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

Some simplifications:

1. Only sender has to adjust.
2. Exact adjustment, i.e. beam is a ray.
3. Fixed positions.



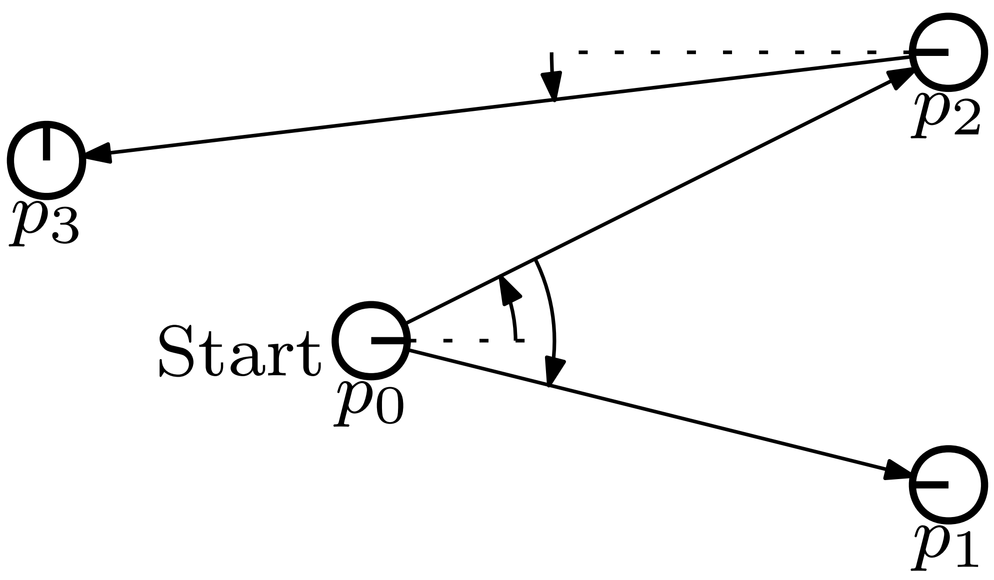
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

Some simplifications:

1. Only sender has to adjust.
2. Exact adjustment, i.e. beam is a ray.
3. Fixed positions.
4. Geometric plane.

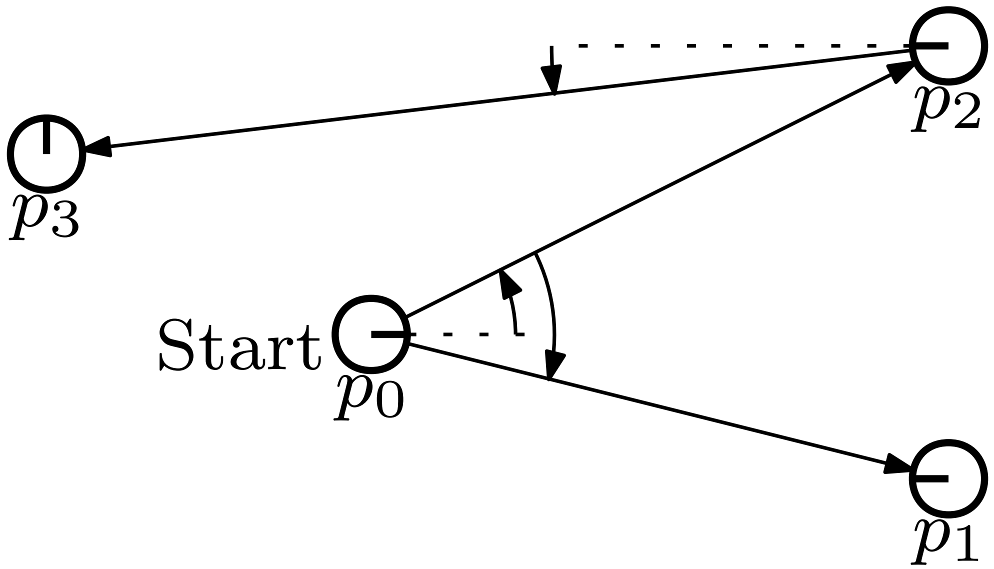


Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

Some simplifications:

1. Only sender has to adjust.
2. Exact adjustment, i.e. beam is a ray.
3. Fixed positions.
4. Geometric plane.
5. Negligible transmission time.



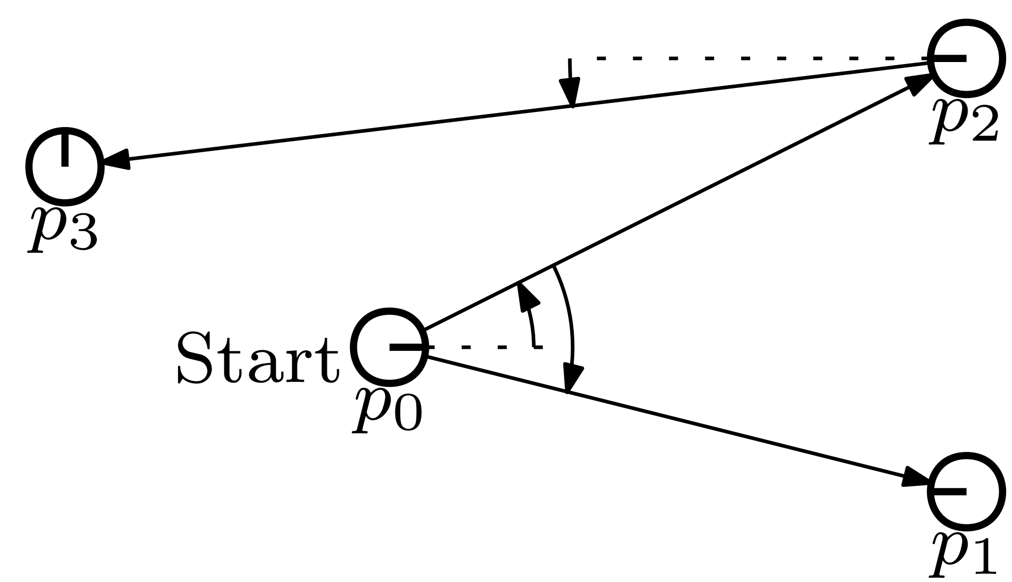
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Problem Description

Distribute data from one satellite to all other with minimal makespan.
Informed/activated satellites can participate in the distribution.

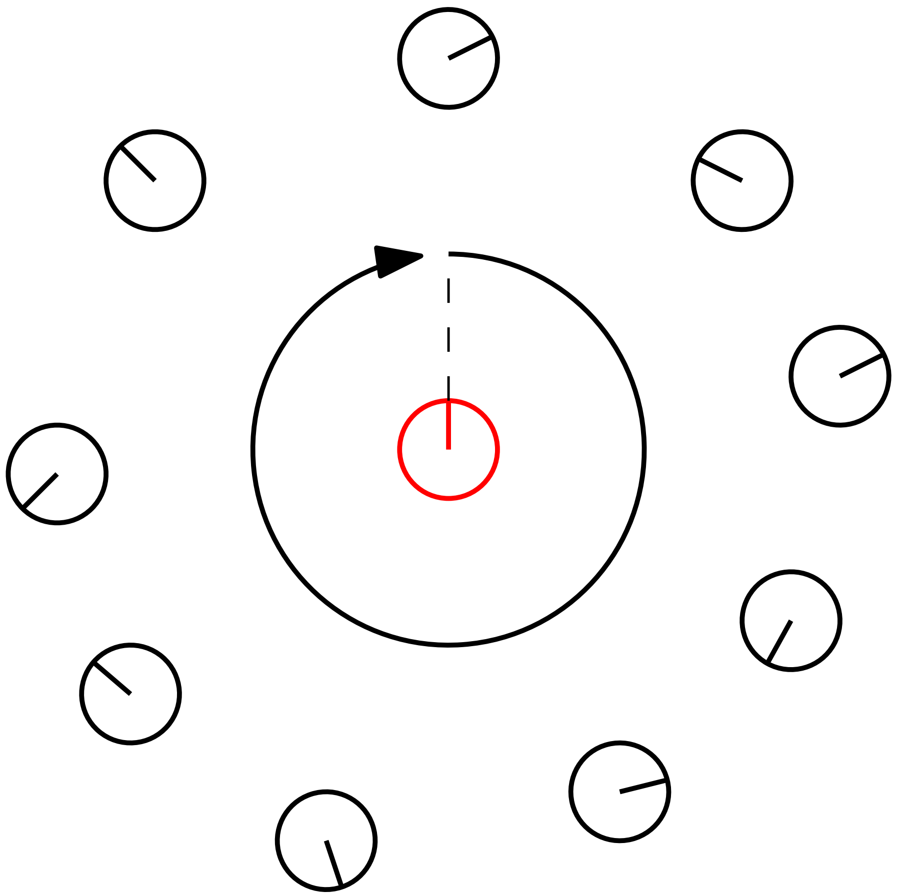
Some simplifications:

1. Only sender has to adjust.
2. Exact adjustment, i.e. beam is a ray.
3. Fixed positions.
4. Geometric plane.
5. Negligible transmission time.
6. Rotation time equal rotation angle.



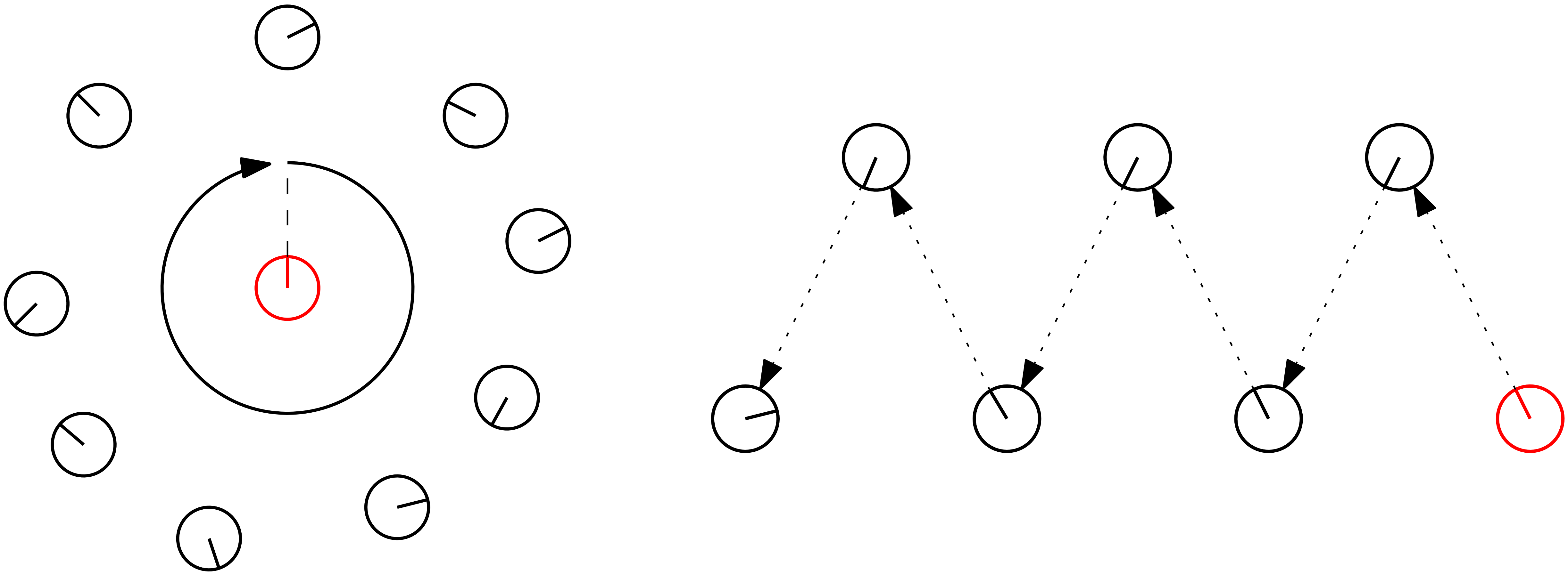
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

A simple observation



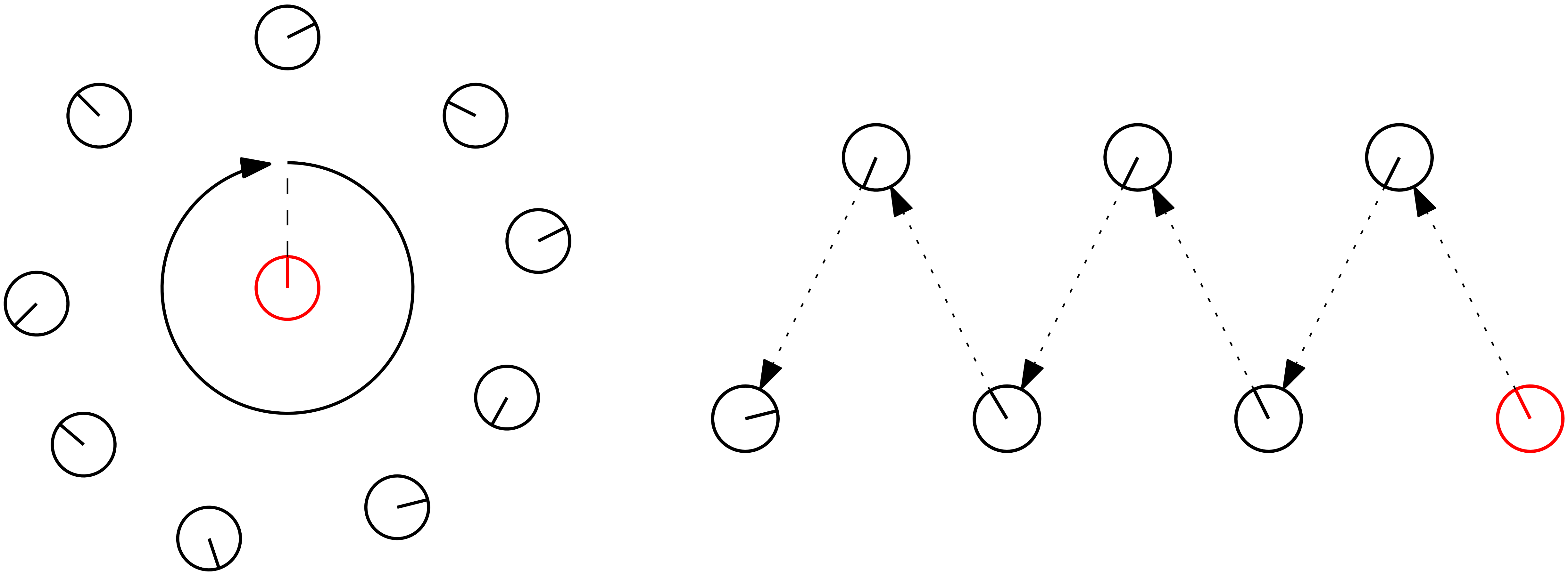
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

A simple observation



Technische Universität Braunschweig
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

A simple observation



Independent of the number of satellites,
the objective value is between 0 and 2π.

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness

Technische Universität Braunschweig
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

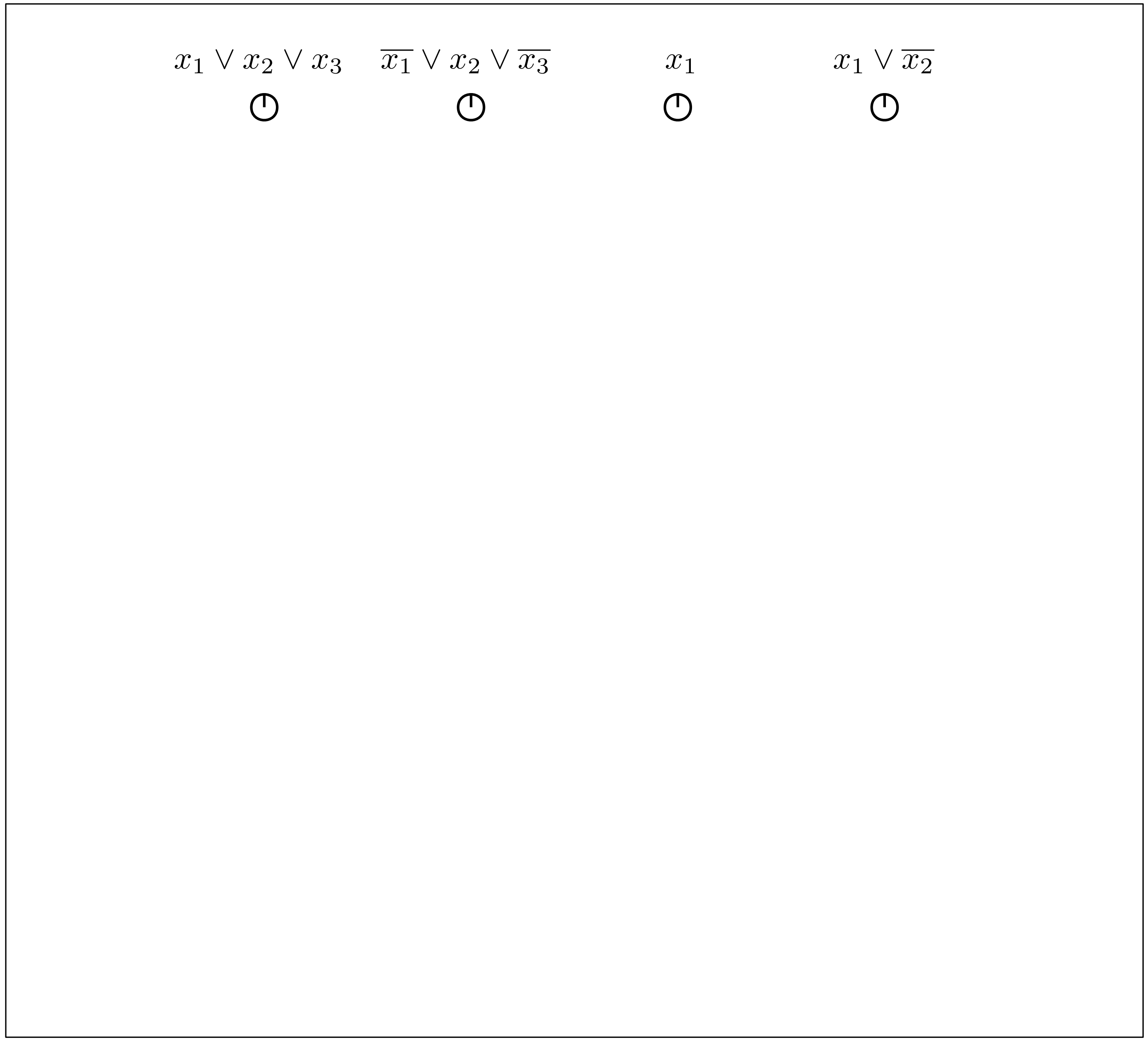
Hardness

Theorem

A c-approximation algorithm for the AFT with $c < 5/3$ implies $P = NP$.

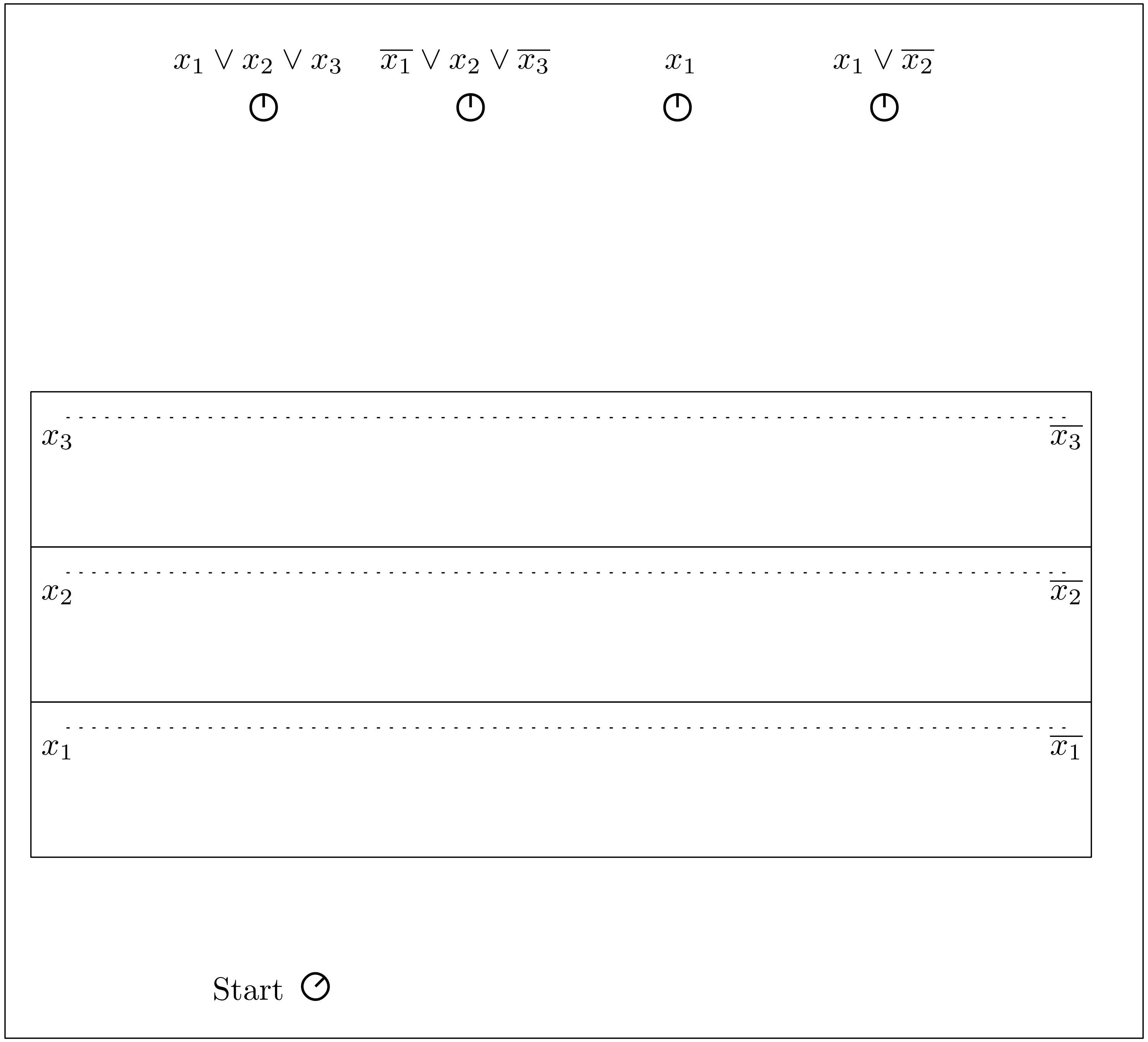
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



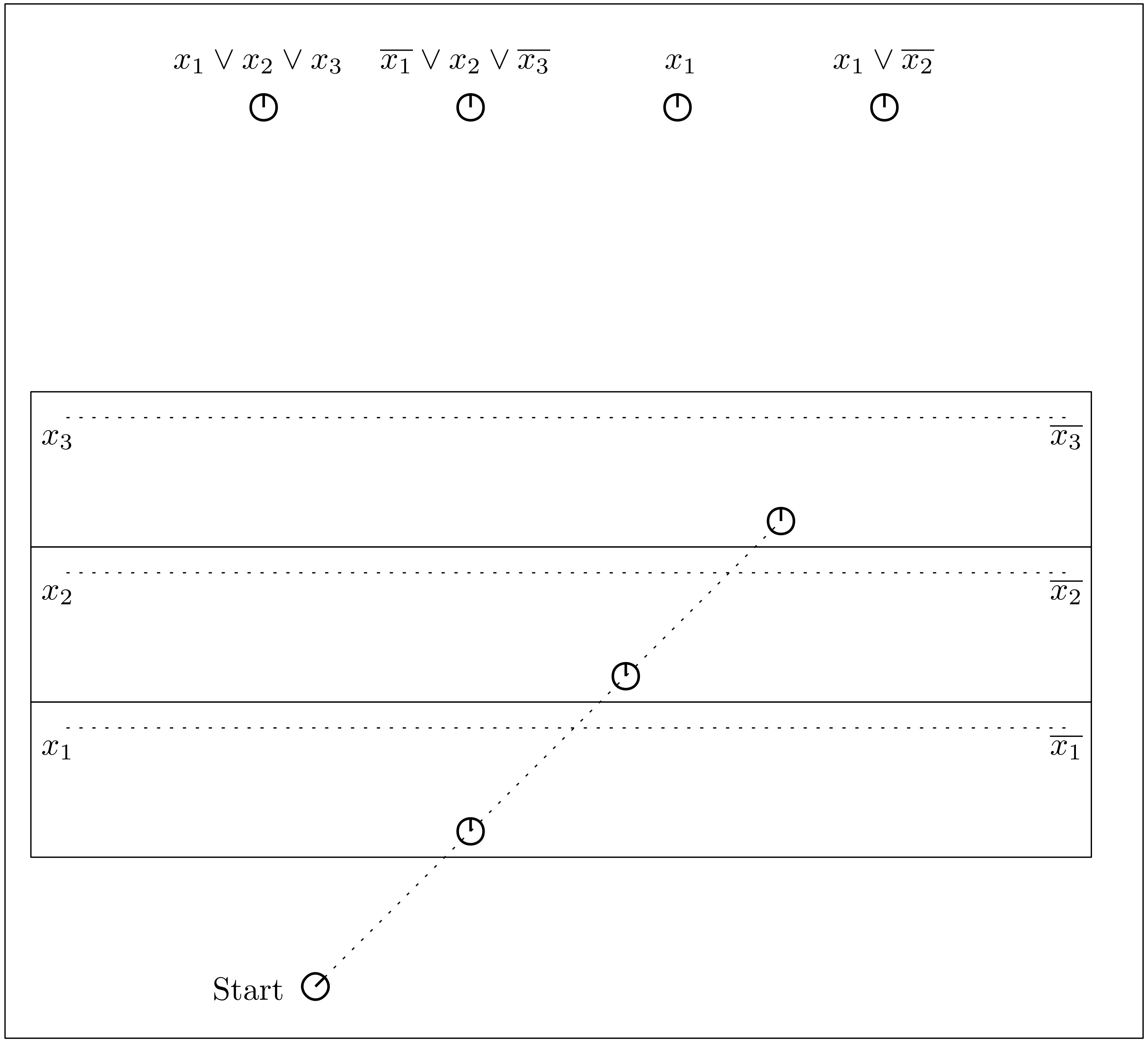
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



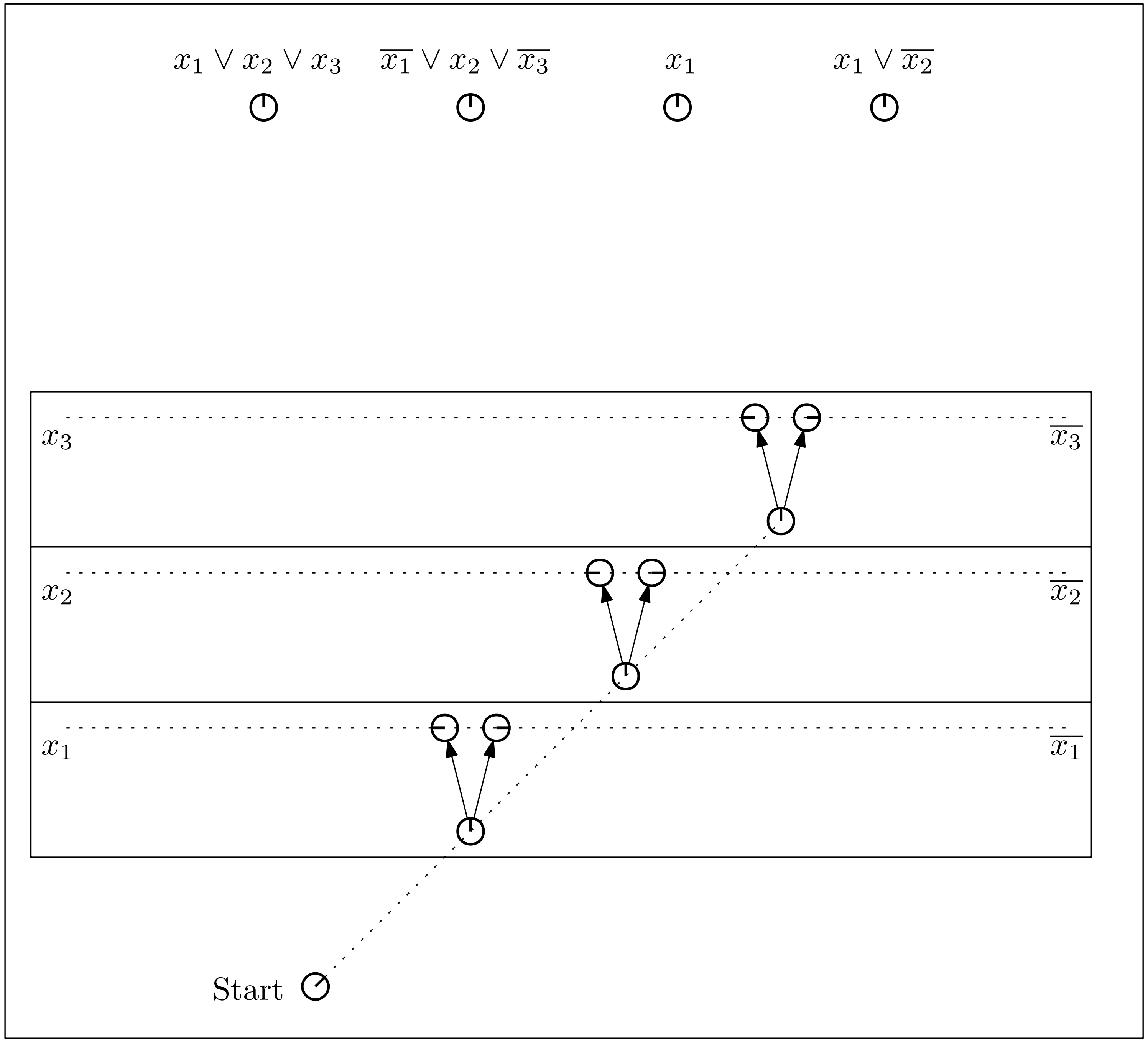
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



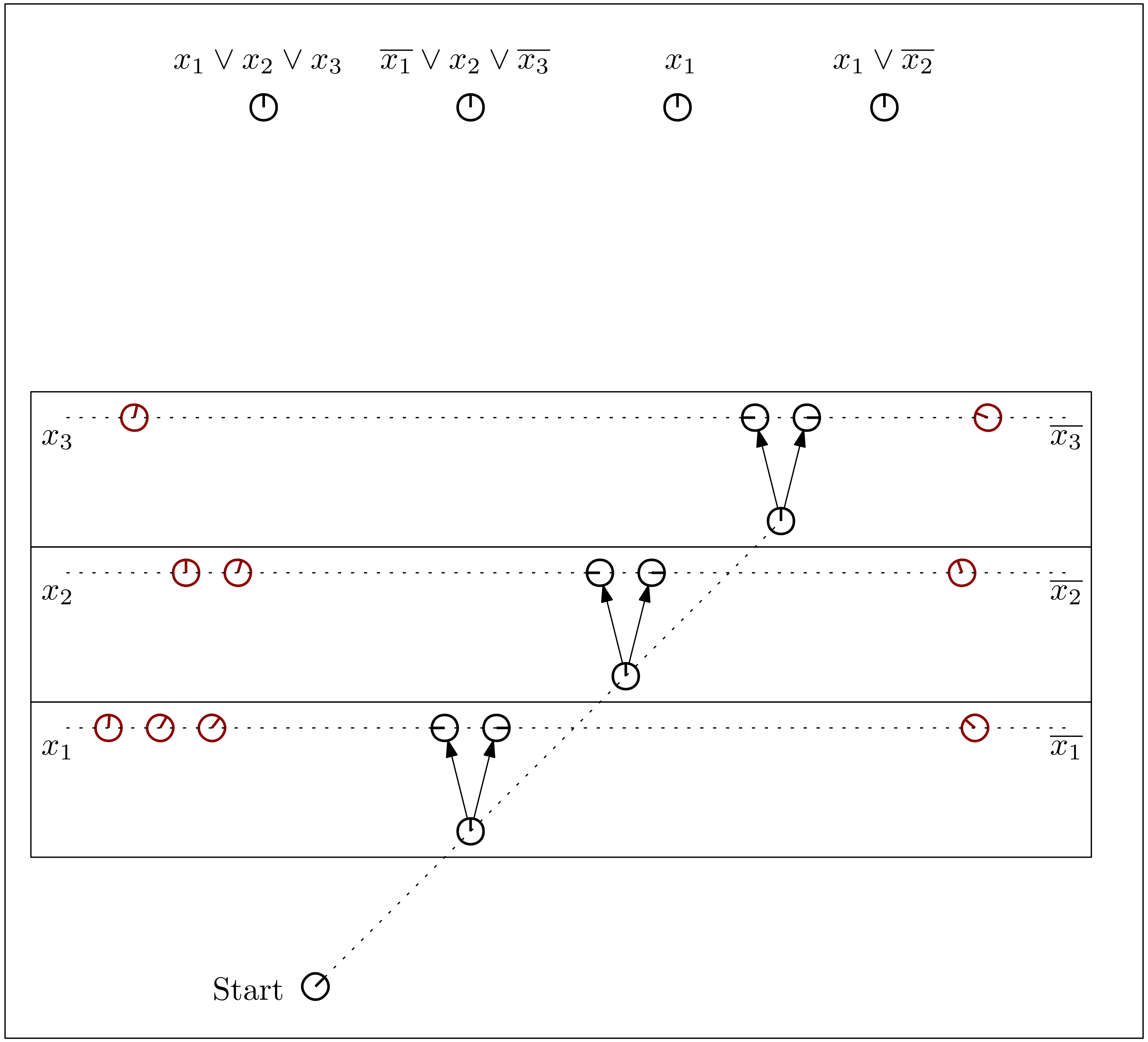
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



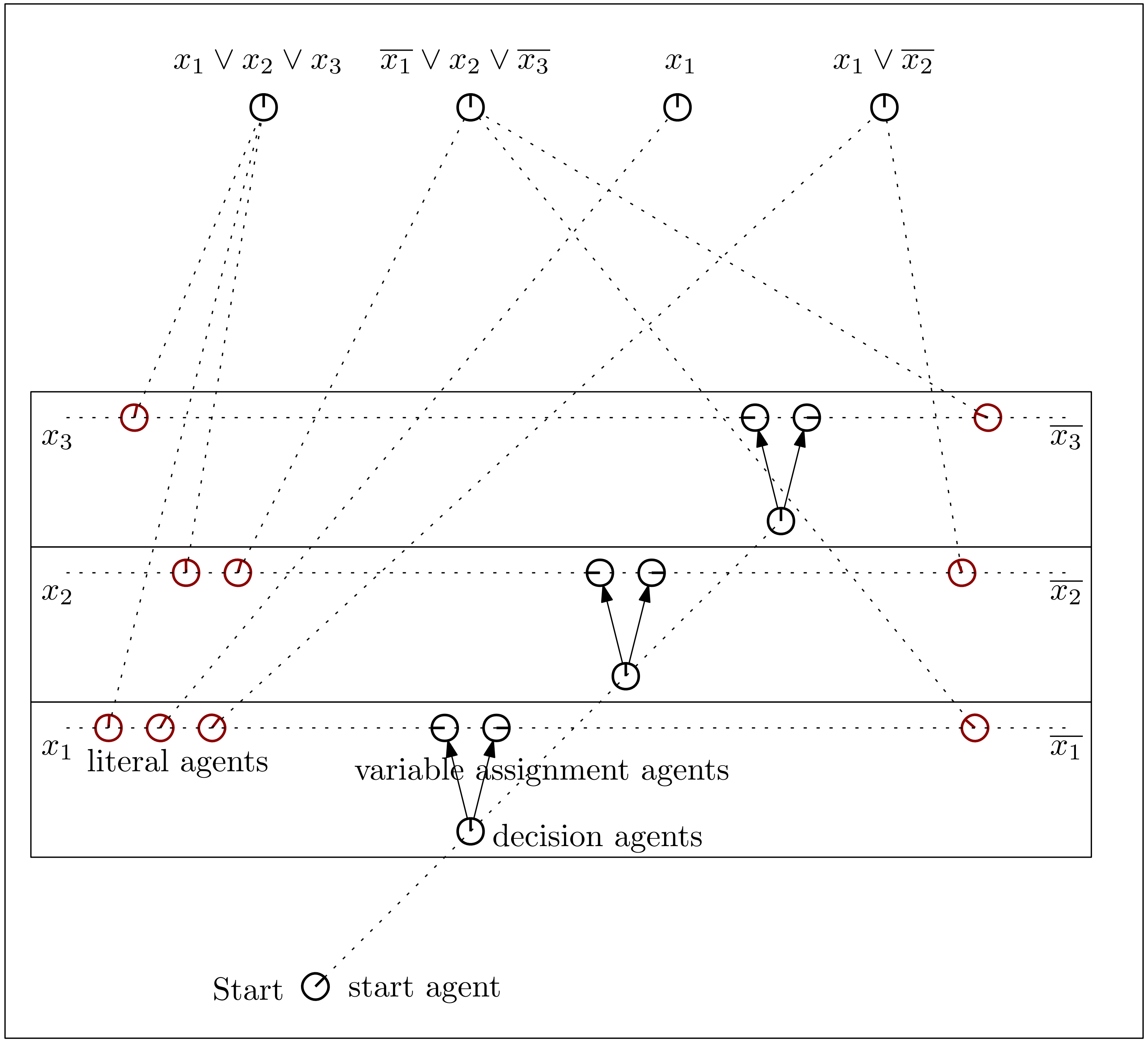
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



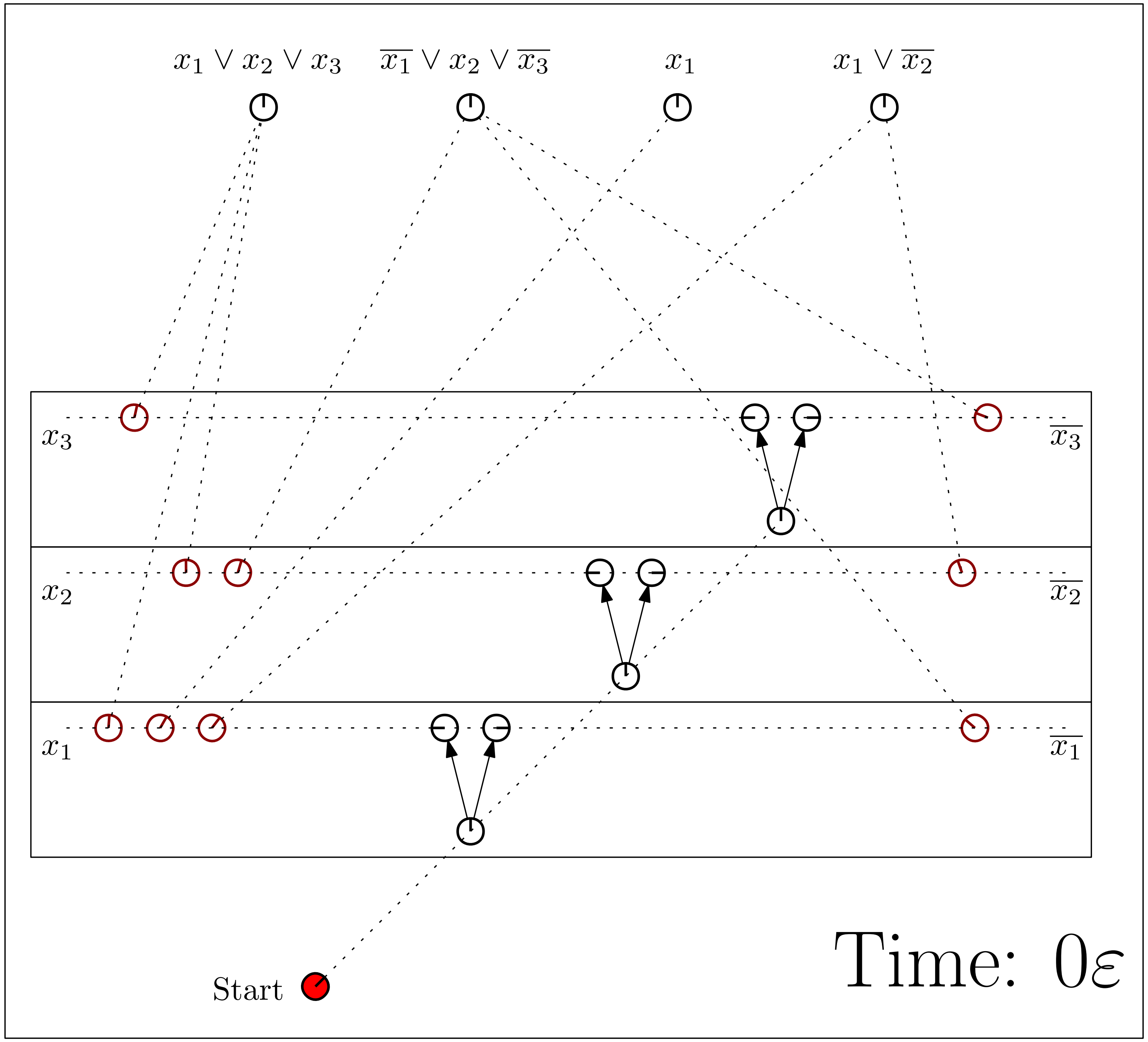
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



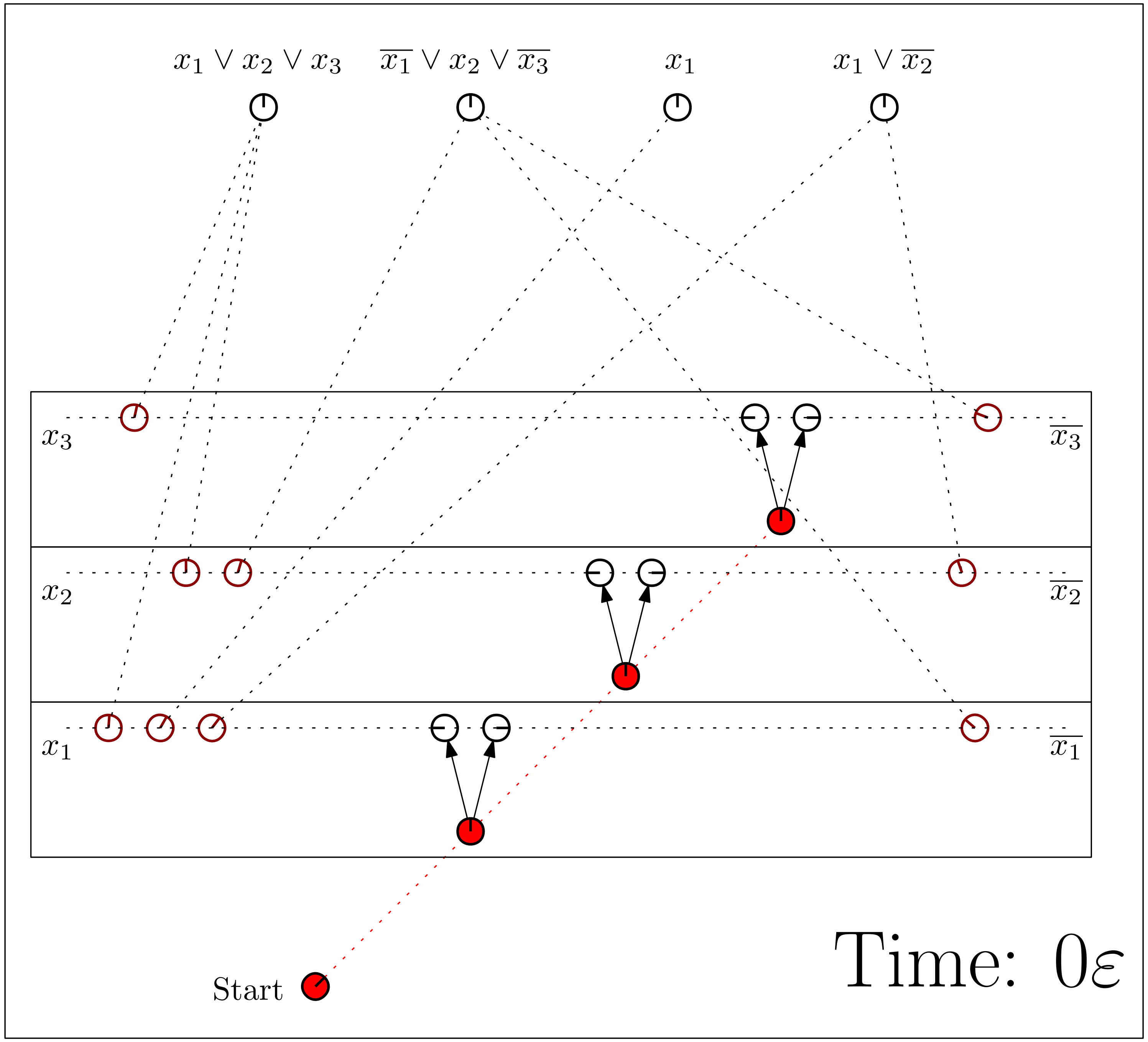
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



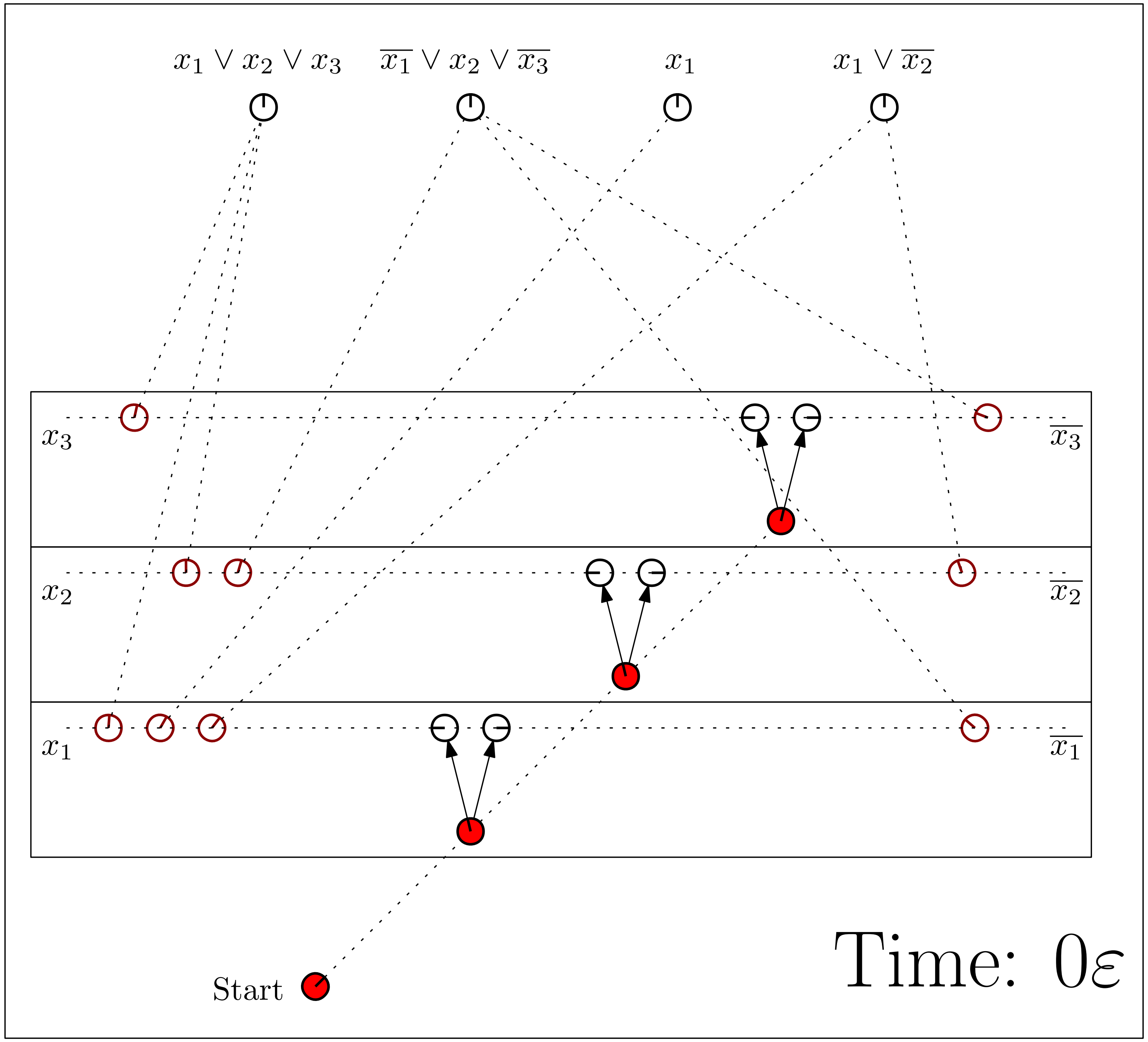
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



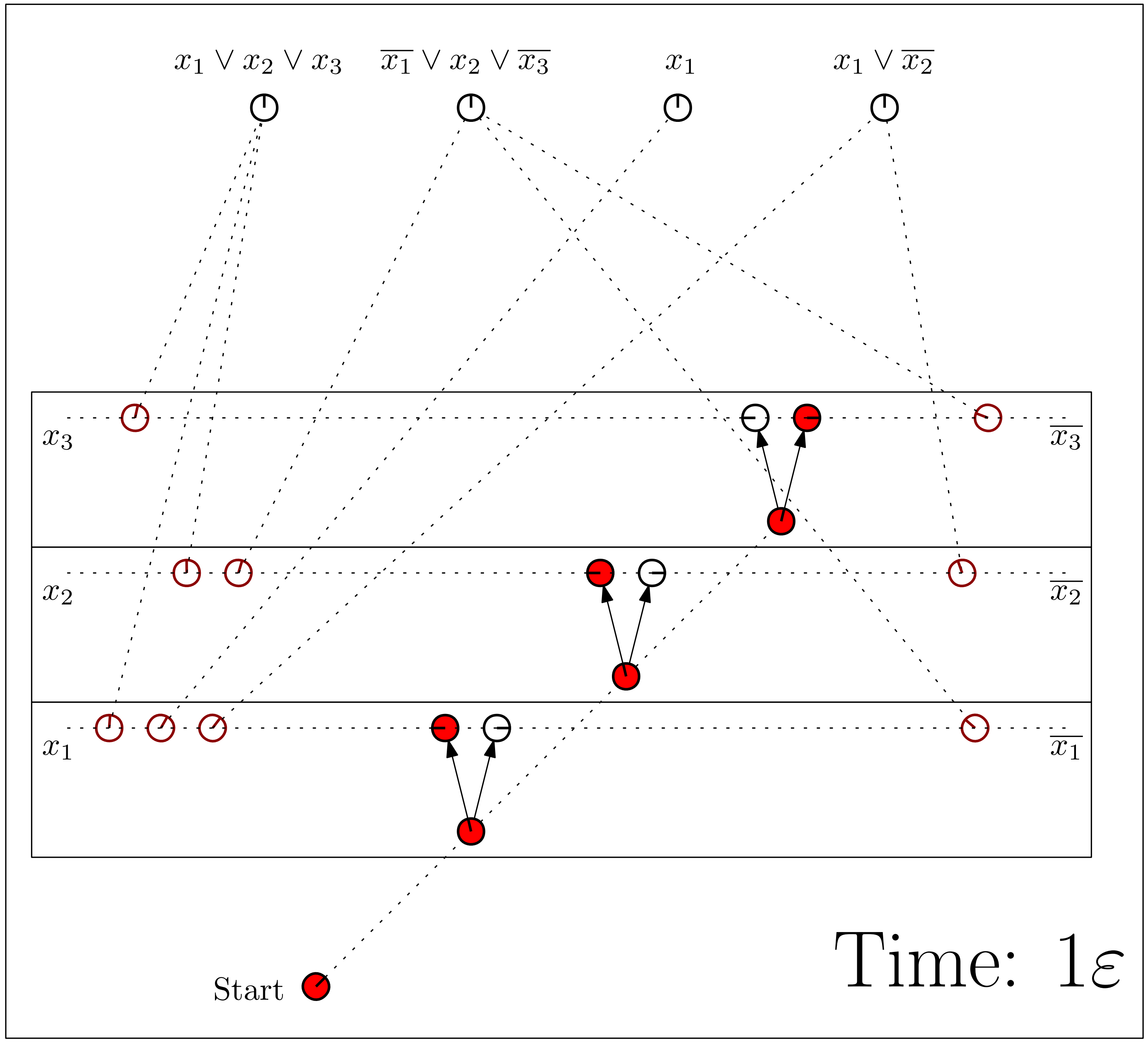
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



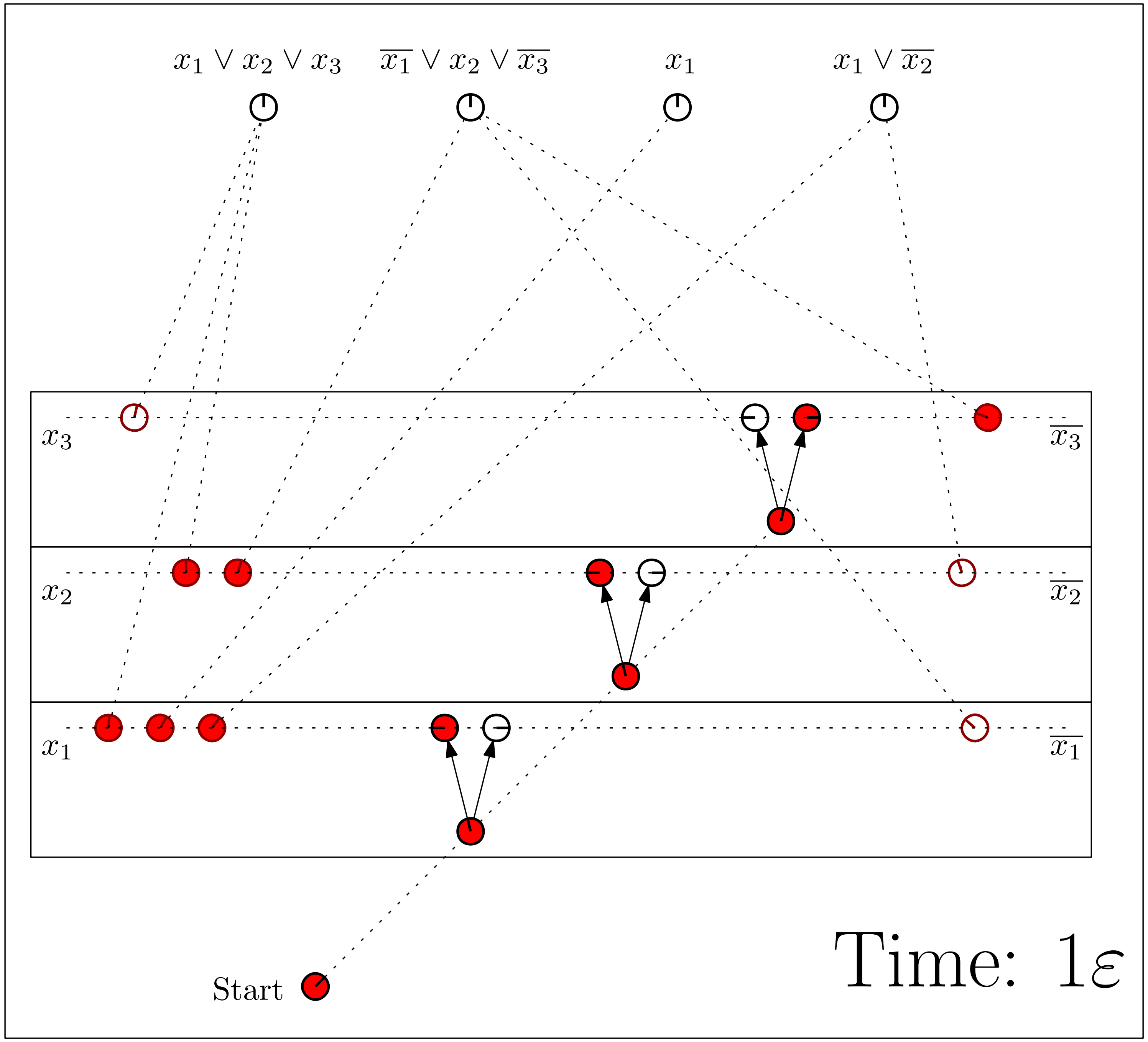
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



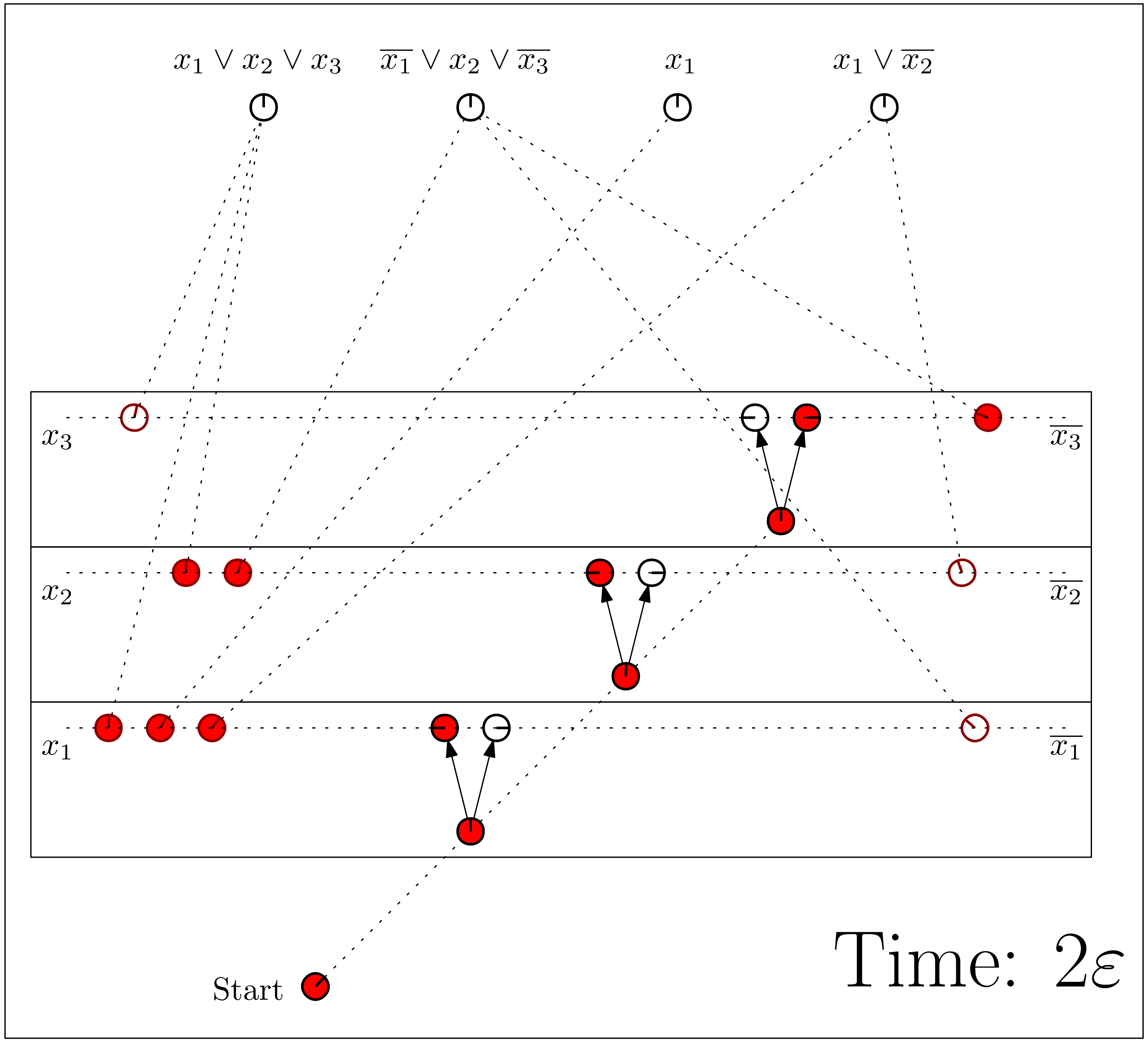
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



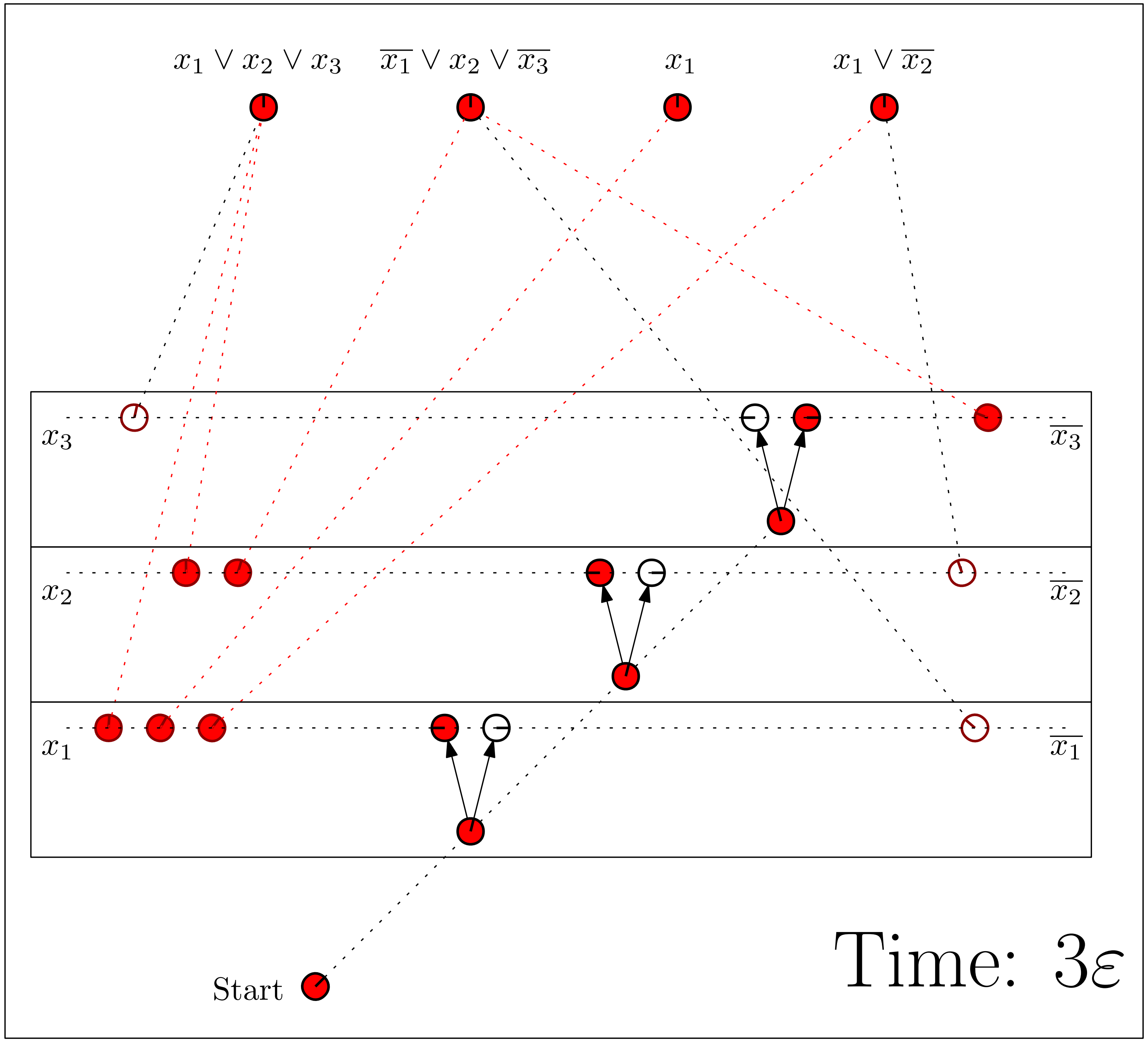
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

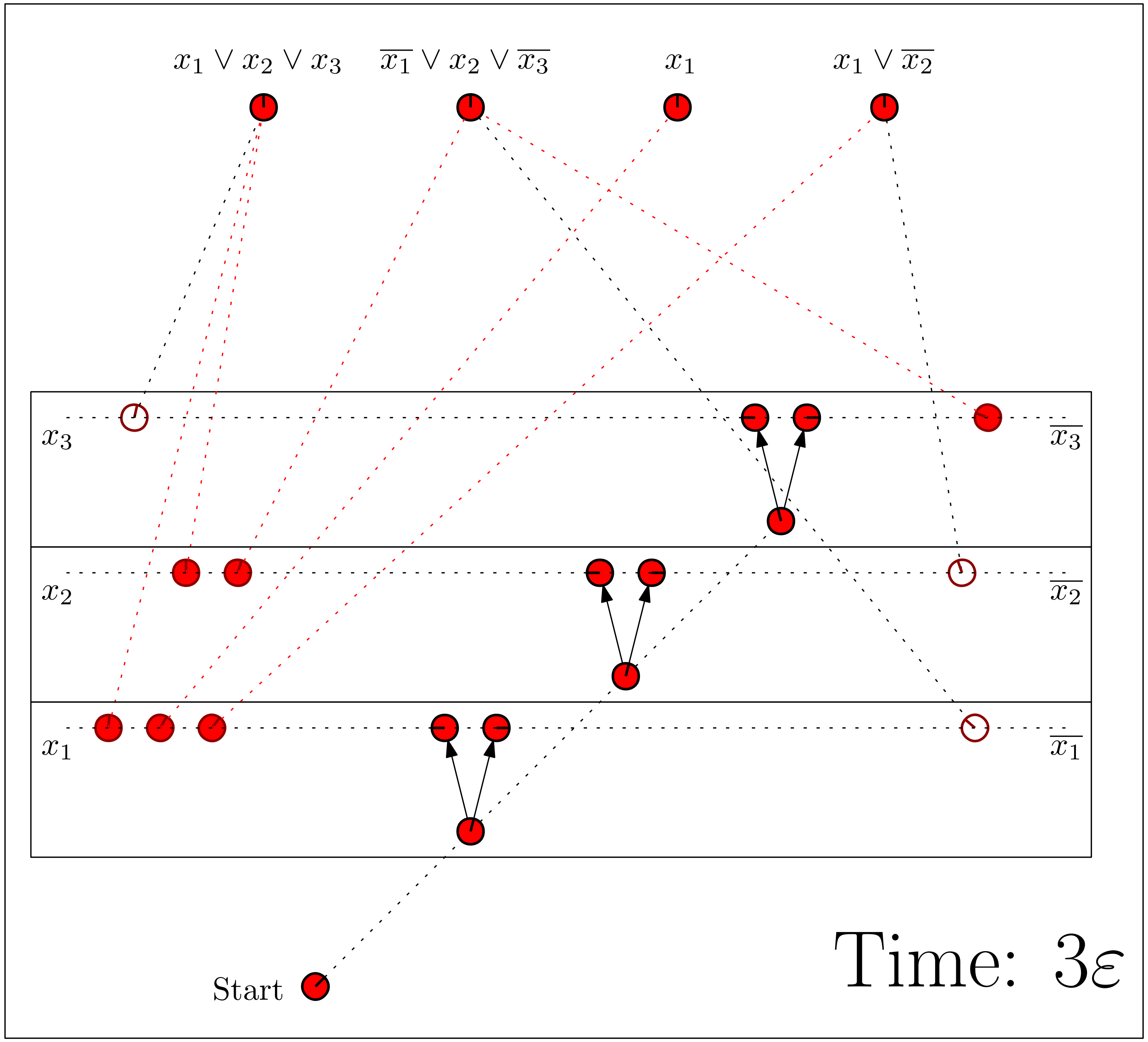
Hardness Construction



d.krupke@tu-bs.de

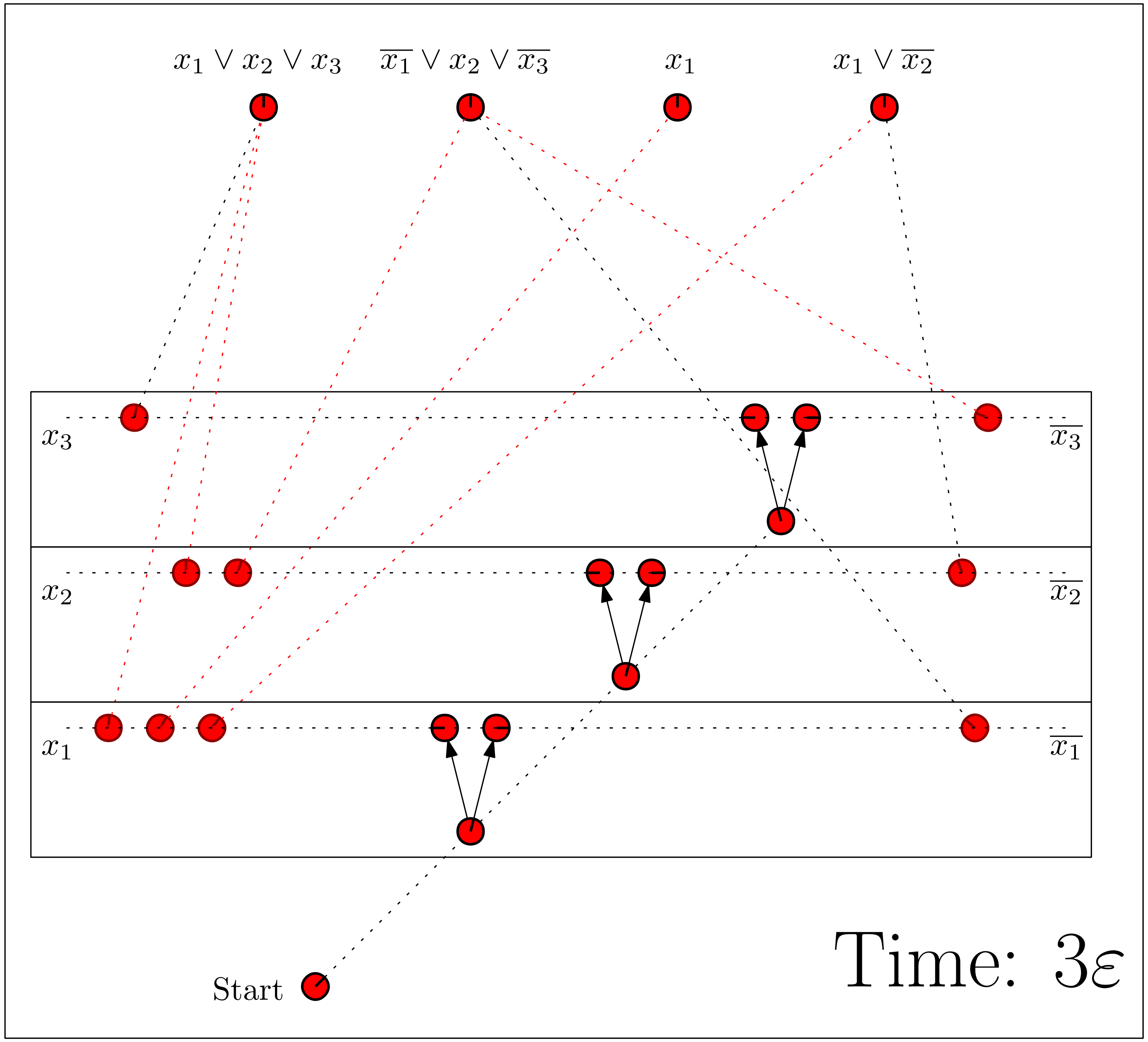
Institute of Operating Systems and Computer Networks

Hardness Construction



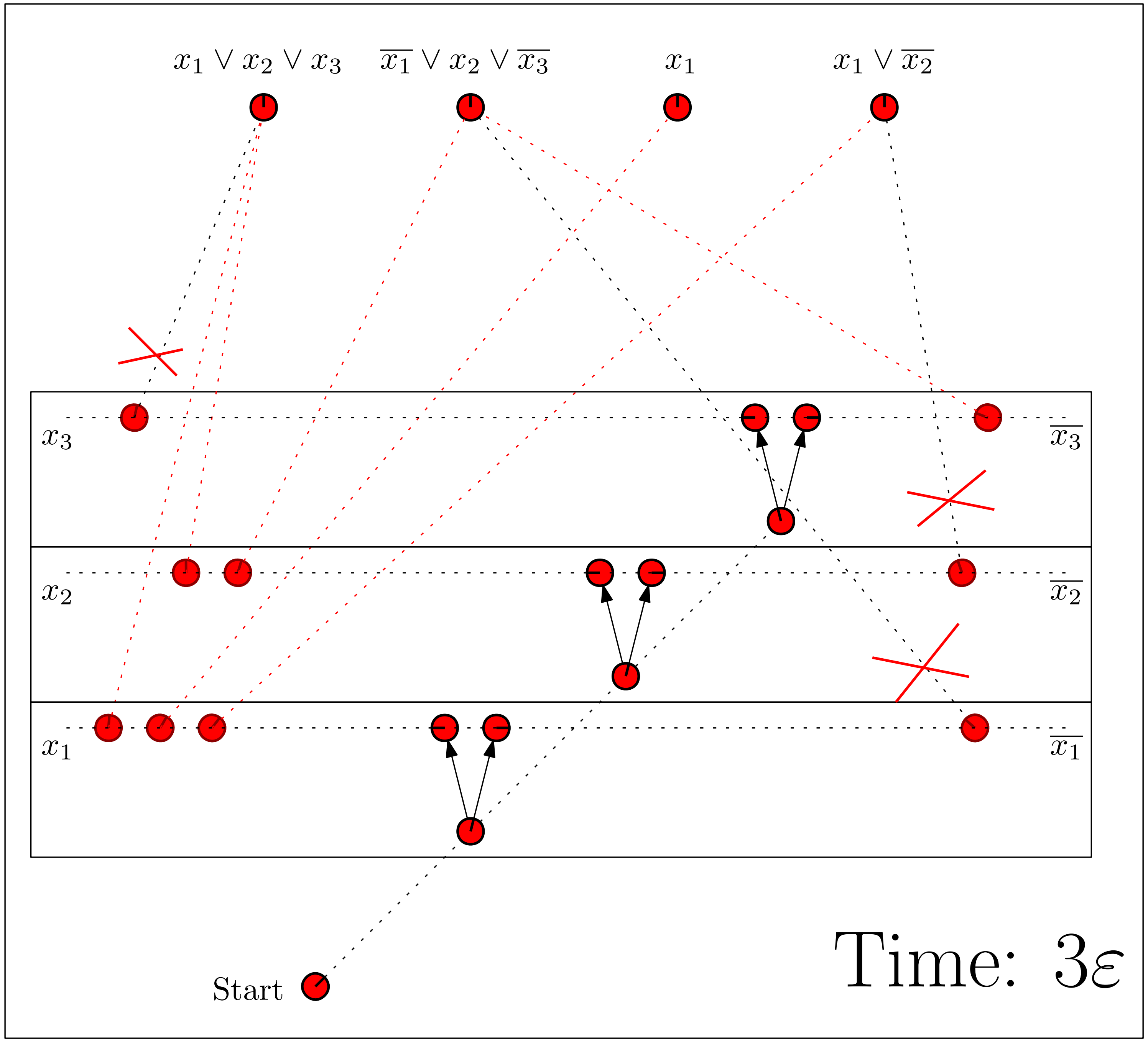
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction



d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Hardness Construction

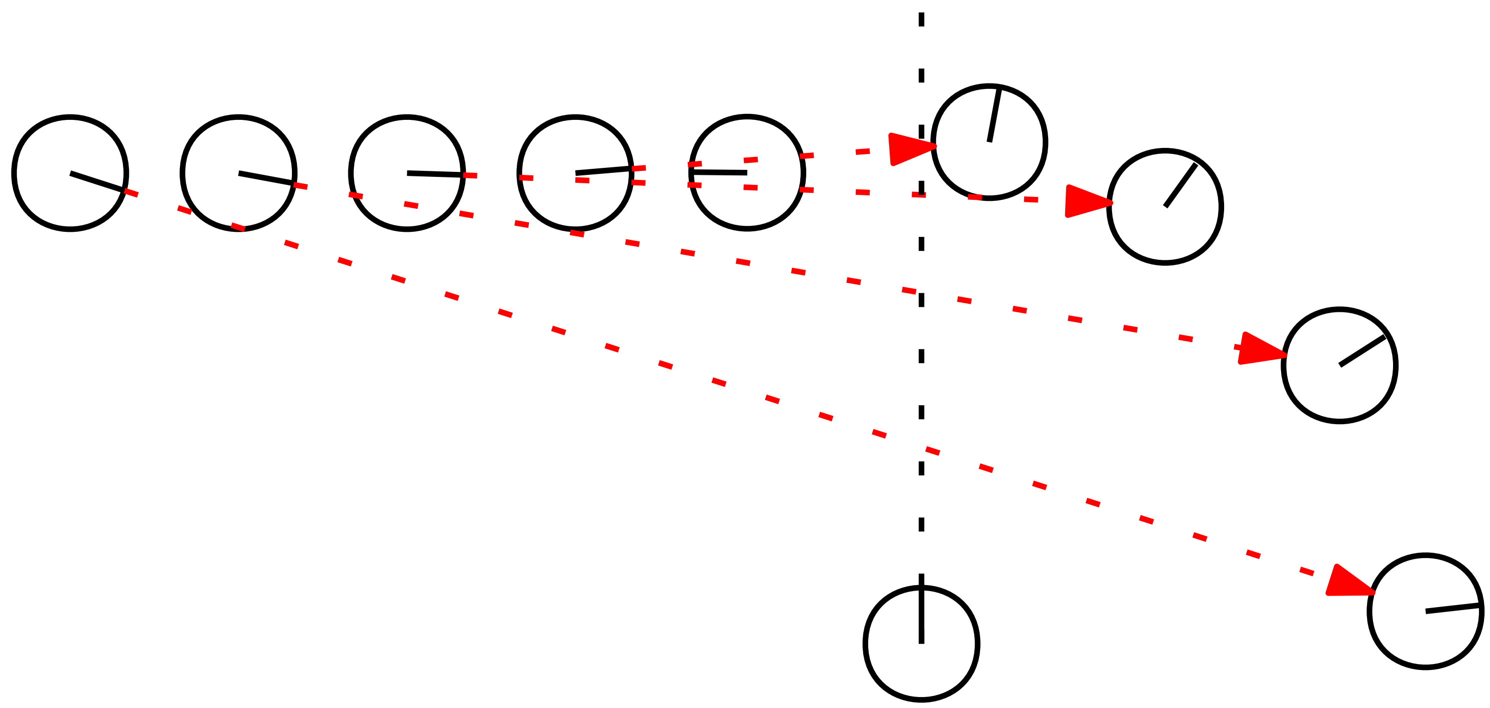


d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Approximation

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

9-Approximation Algorithm



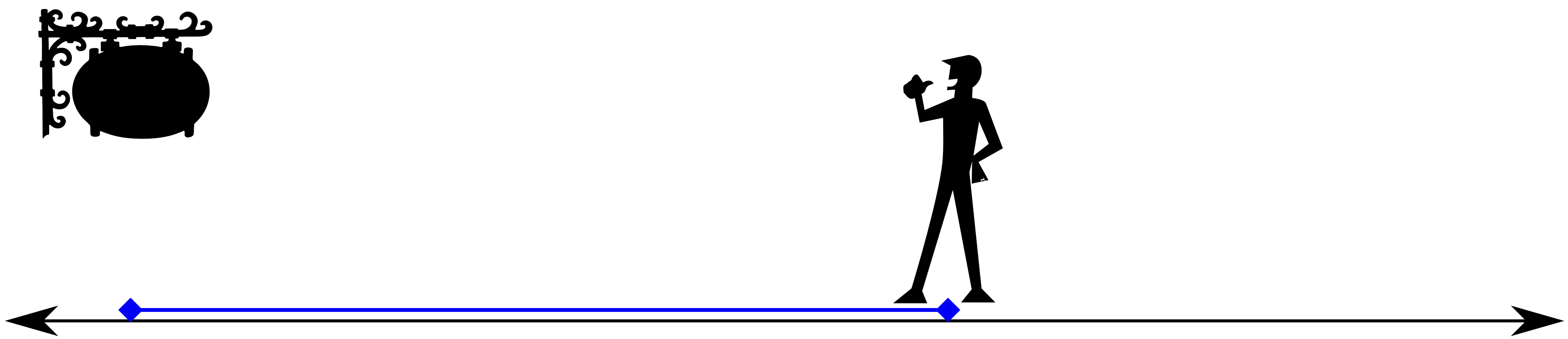
Theorem

There is a 9-approximation in the plane

Even for unknown locations and headings as long as we know a lower bound of $\varepsilon > 0$

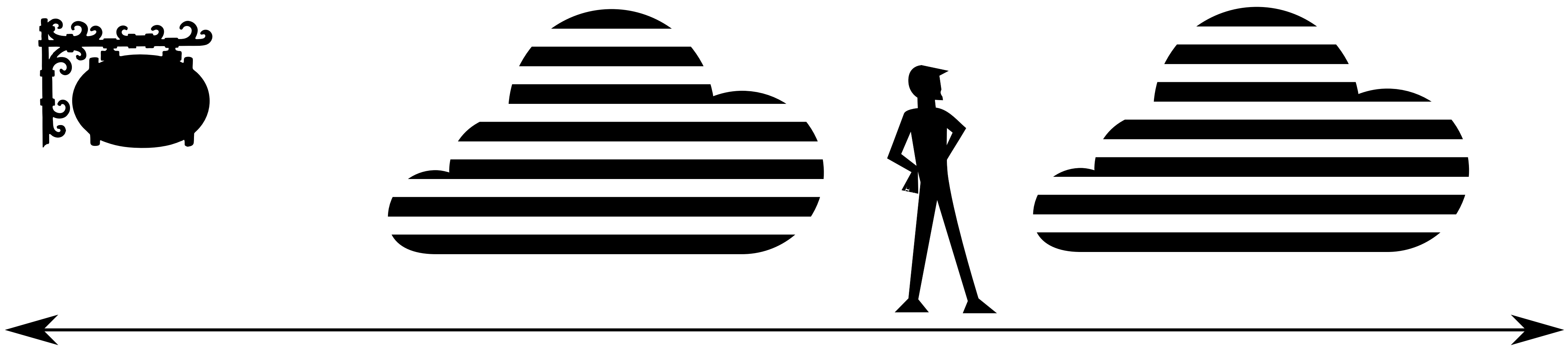
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



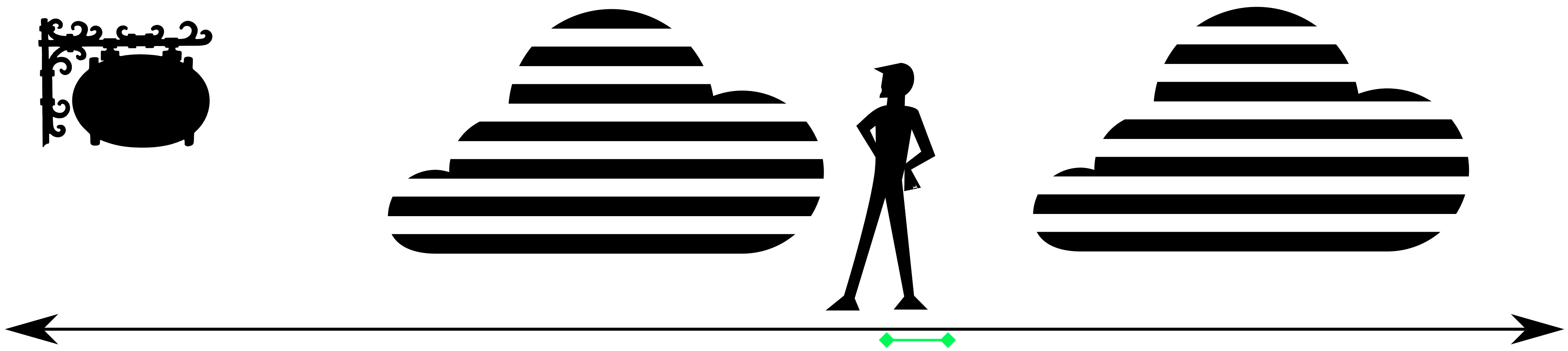
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



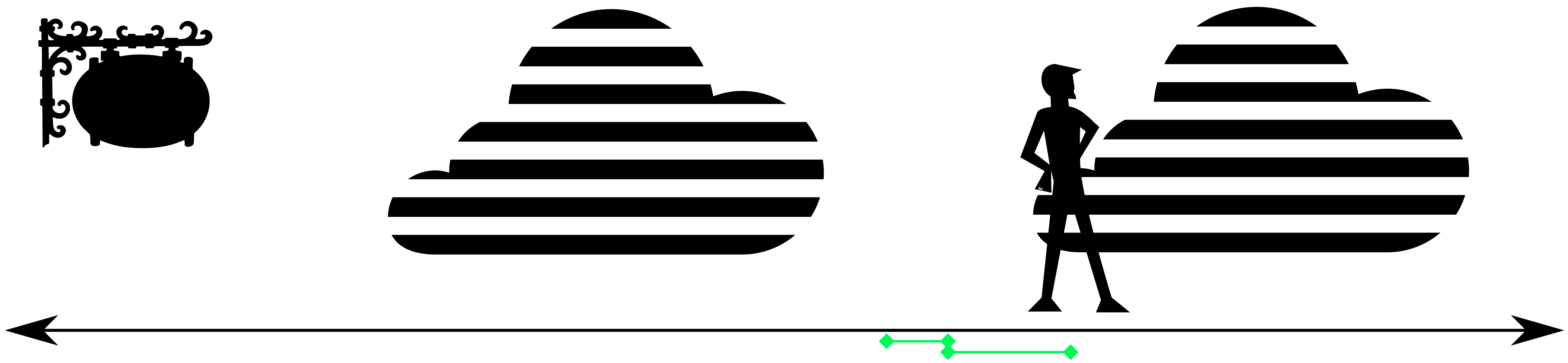
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



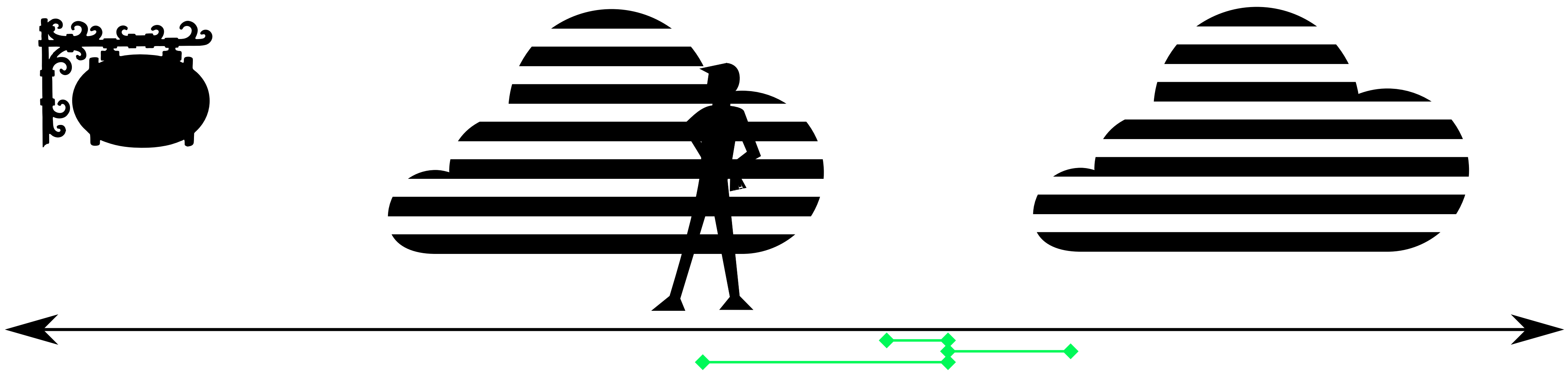
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



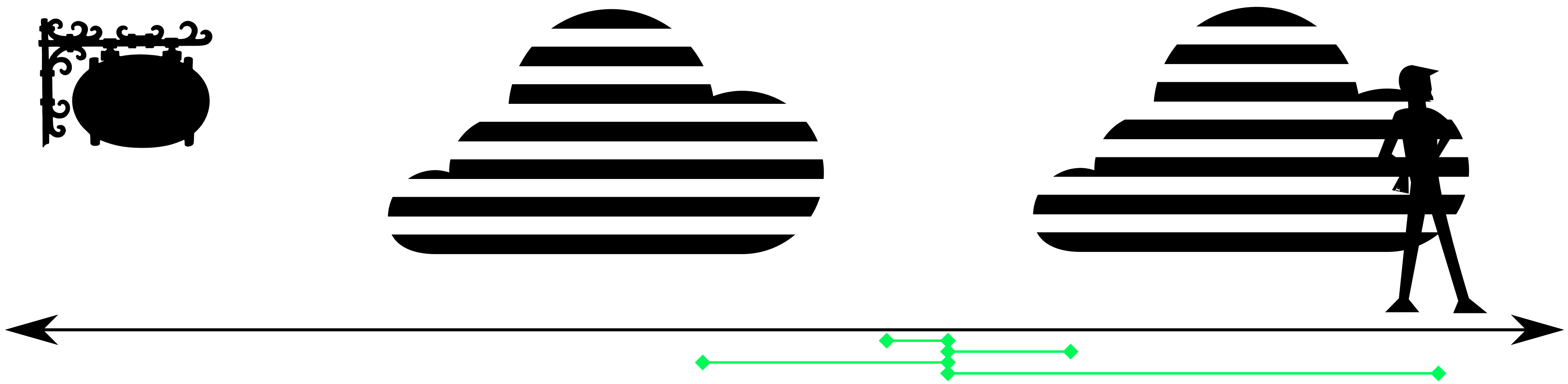
Technische Universität Braunschweig
d.krupke@tu-bs.de
Institute of Operating Systems
and Computer Networks

Recap: The linear search problem



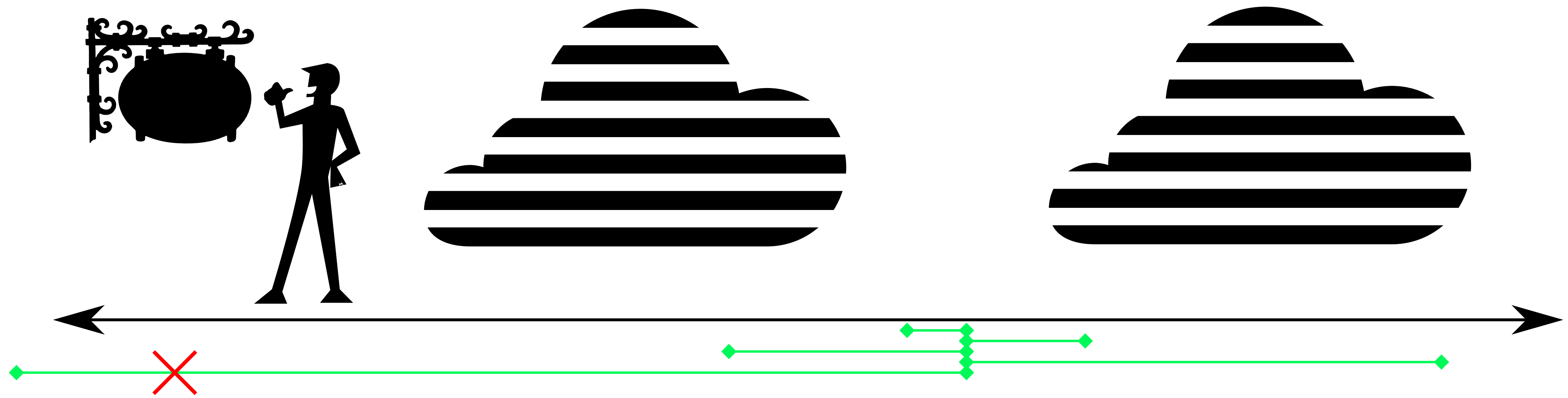
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



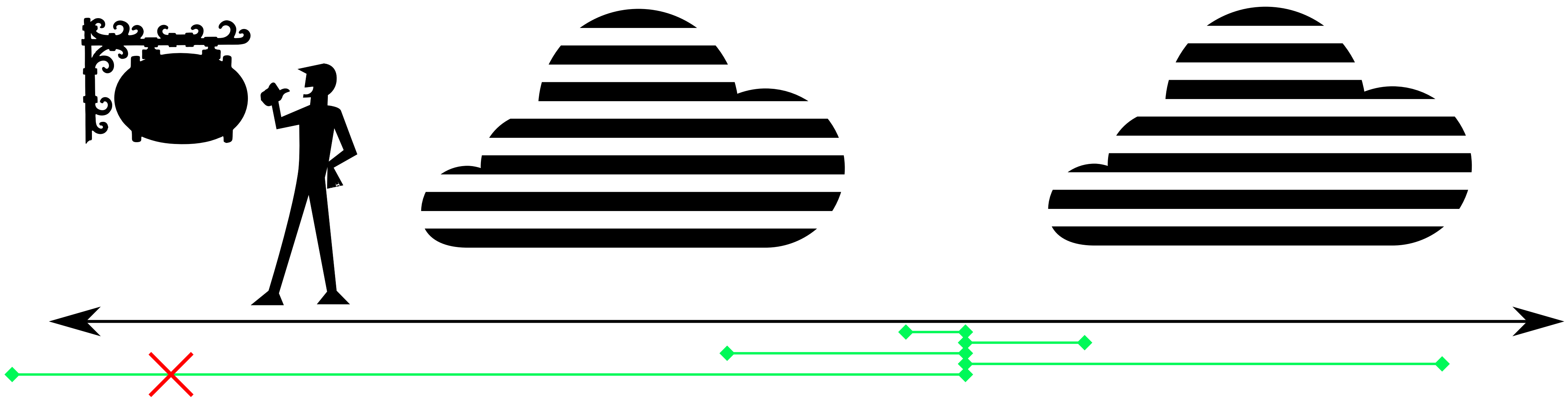
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Recap: The linear search problem



Theorem (Beck and Newman, 1970)

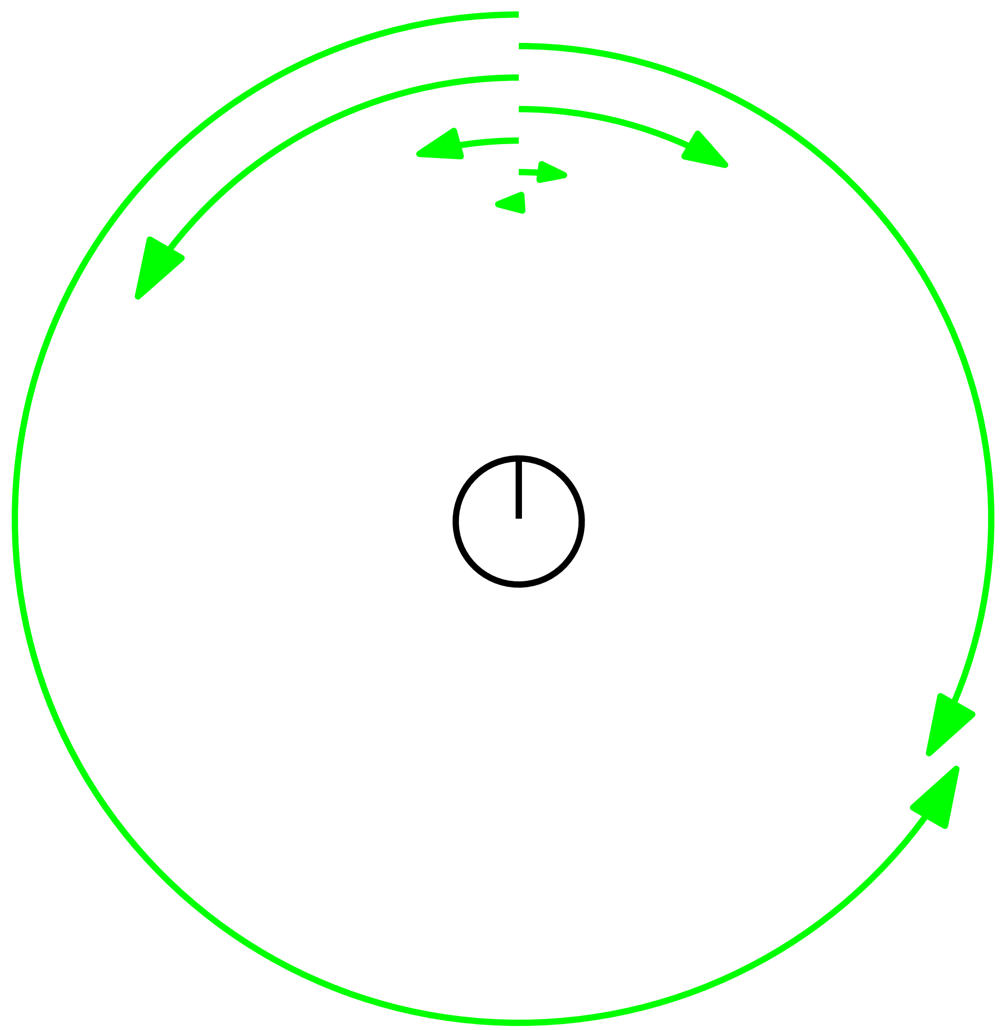
The path of the doubling strategy is at most 9 times the length of the direct path.

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Rotating Linear Search

Strategy

For each satellite: As soon as activated, start doubling rotation.

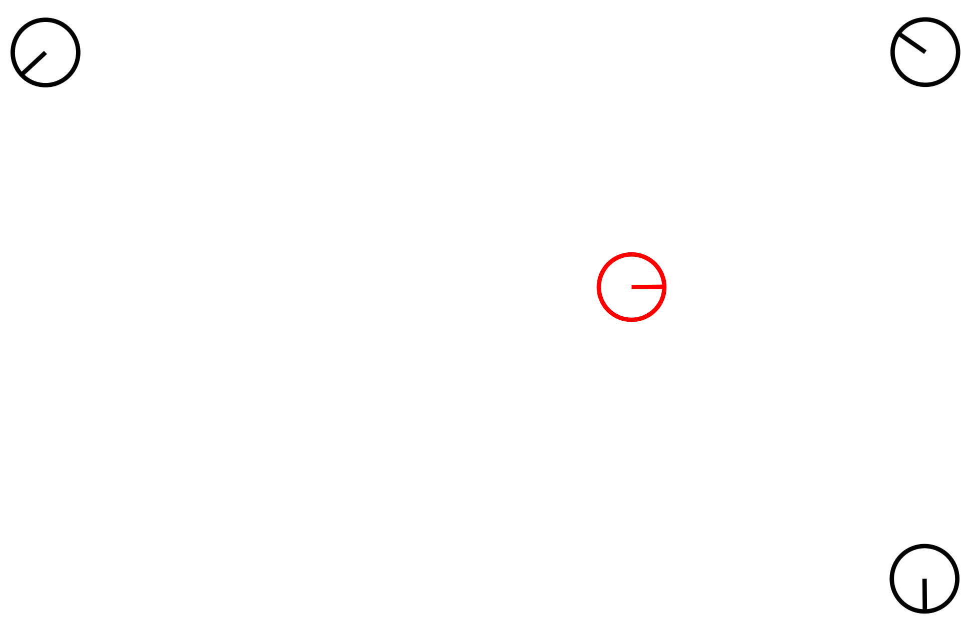


d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming

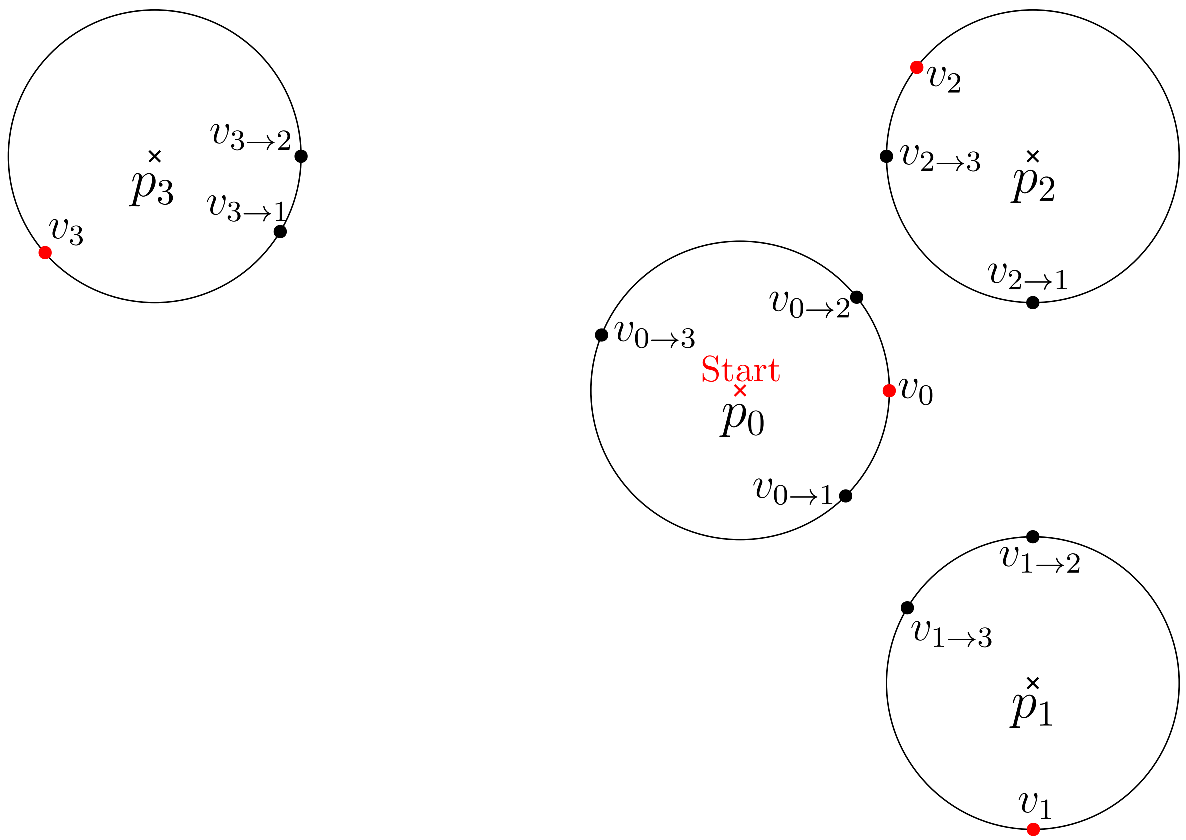


d.krupke@tu-bs.de

Technische Universität Braunschweig

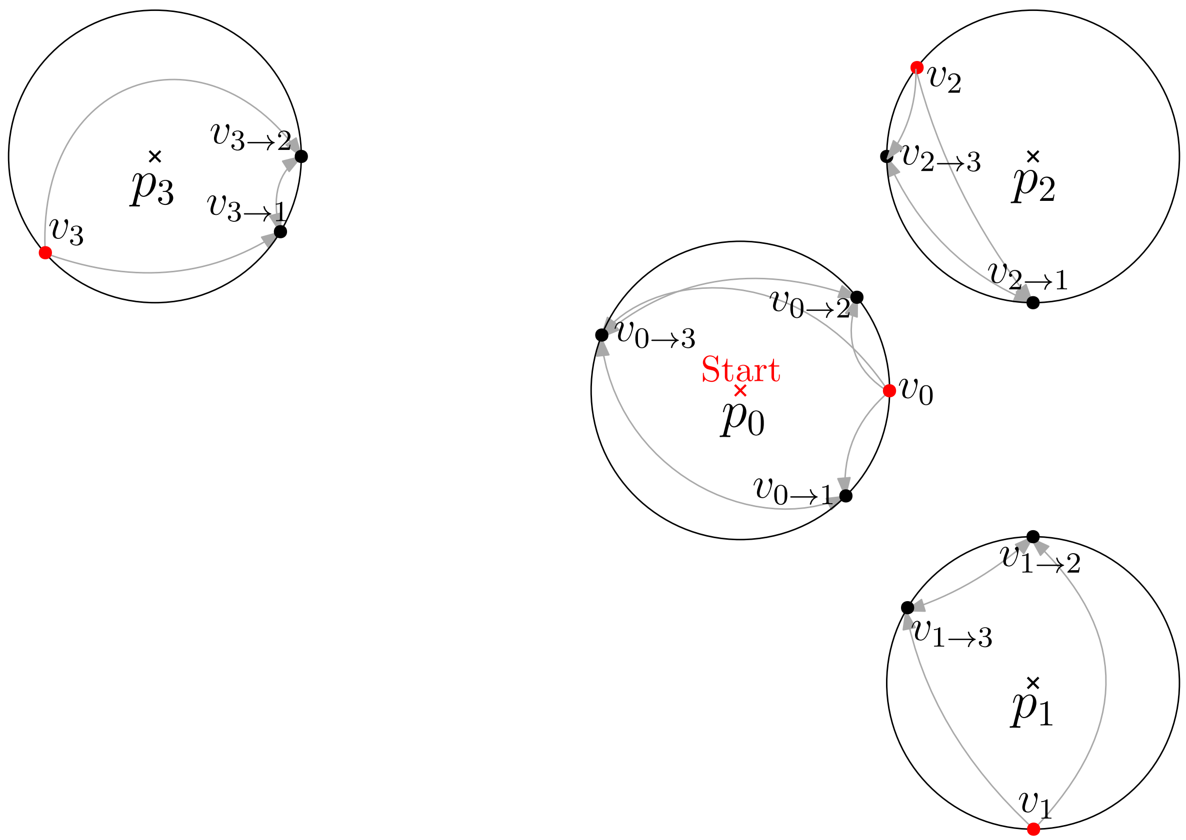
Institute of Operating Systems and Computer Networks

Integer Programming



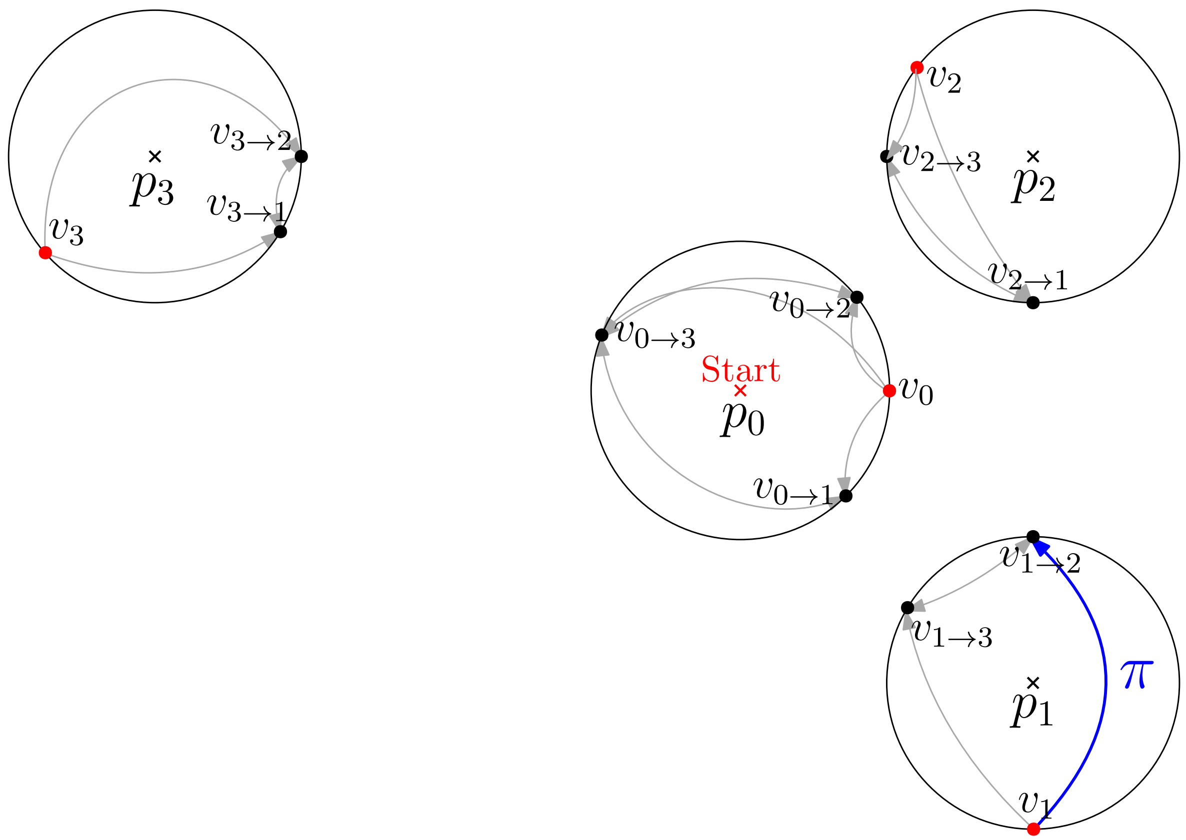
d.krupke@tu-bs.de
Institute of Operating Systems
and Computer Networks

Integer Programming



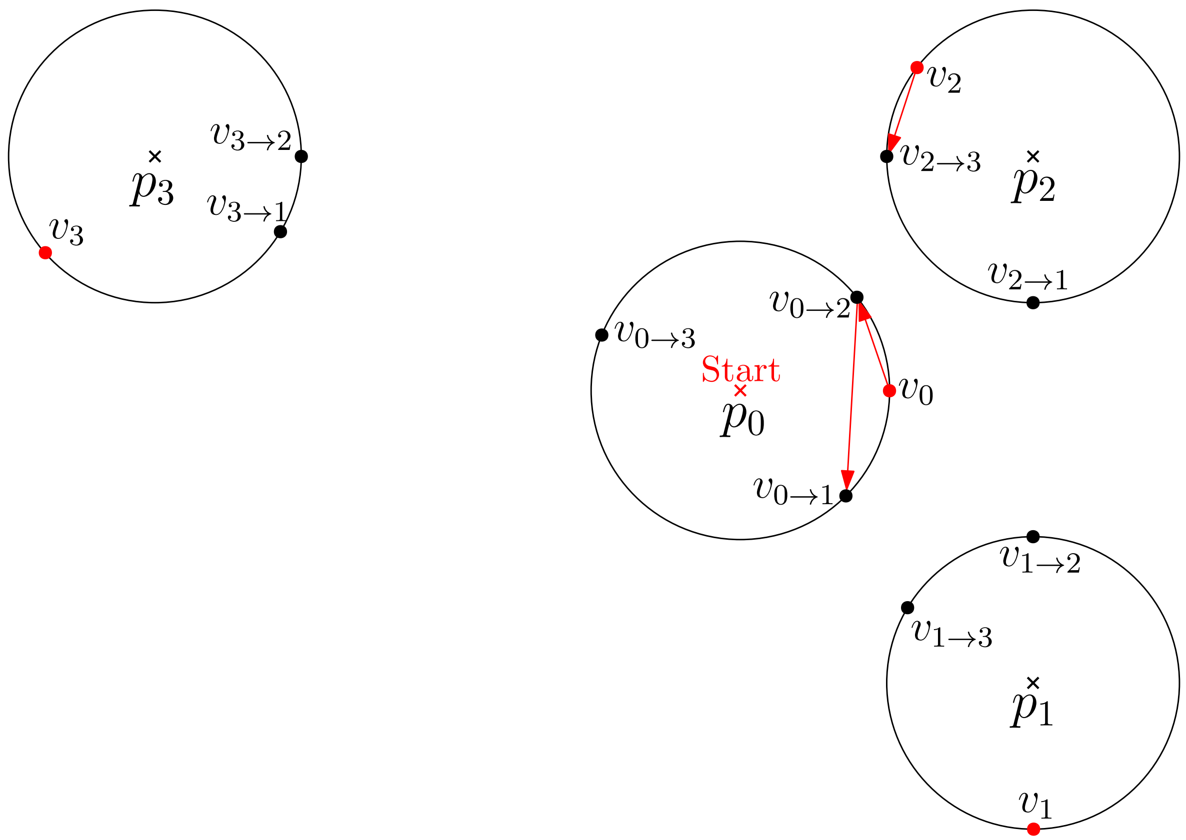
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



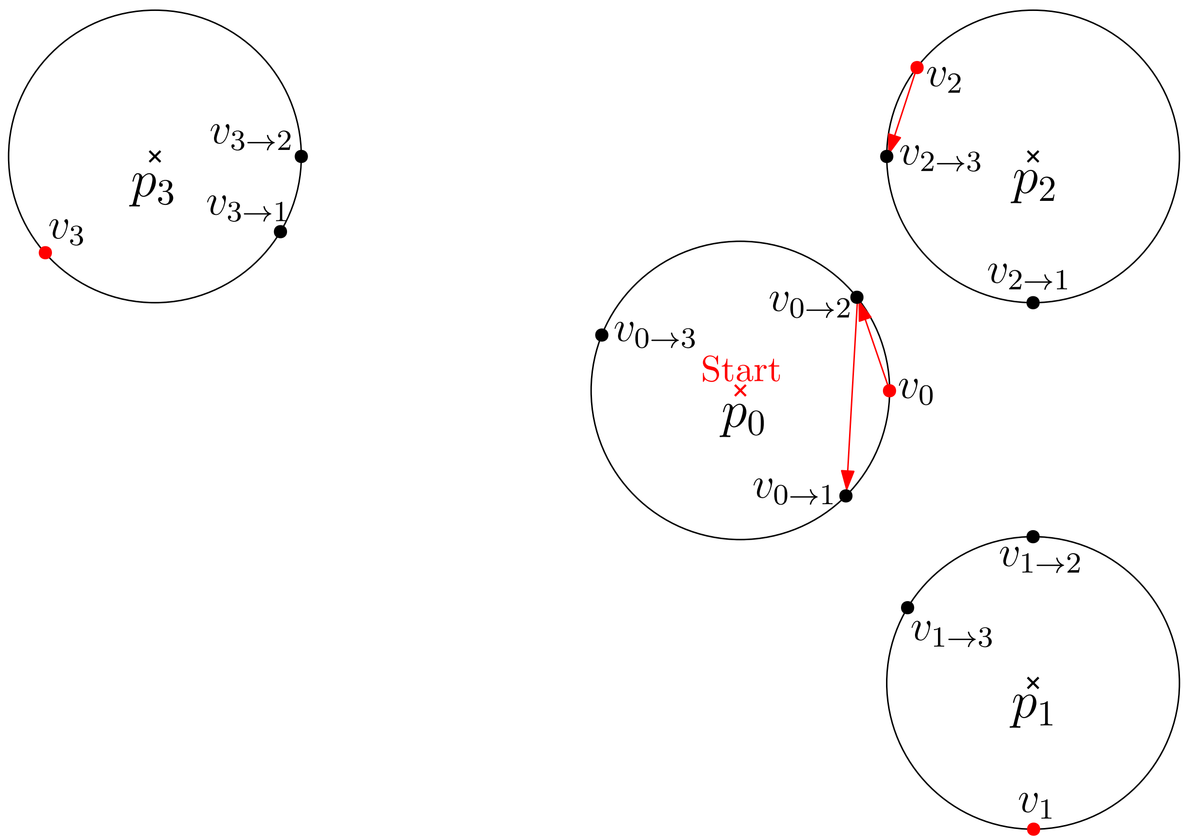
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \leq \sum_{E_{\text{in}}(v_{i\to j})} x_e \leq 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \leq 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \leq |S| - 1 \qquad \forall S \subset V$$

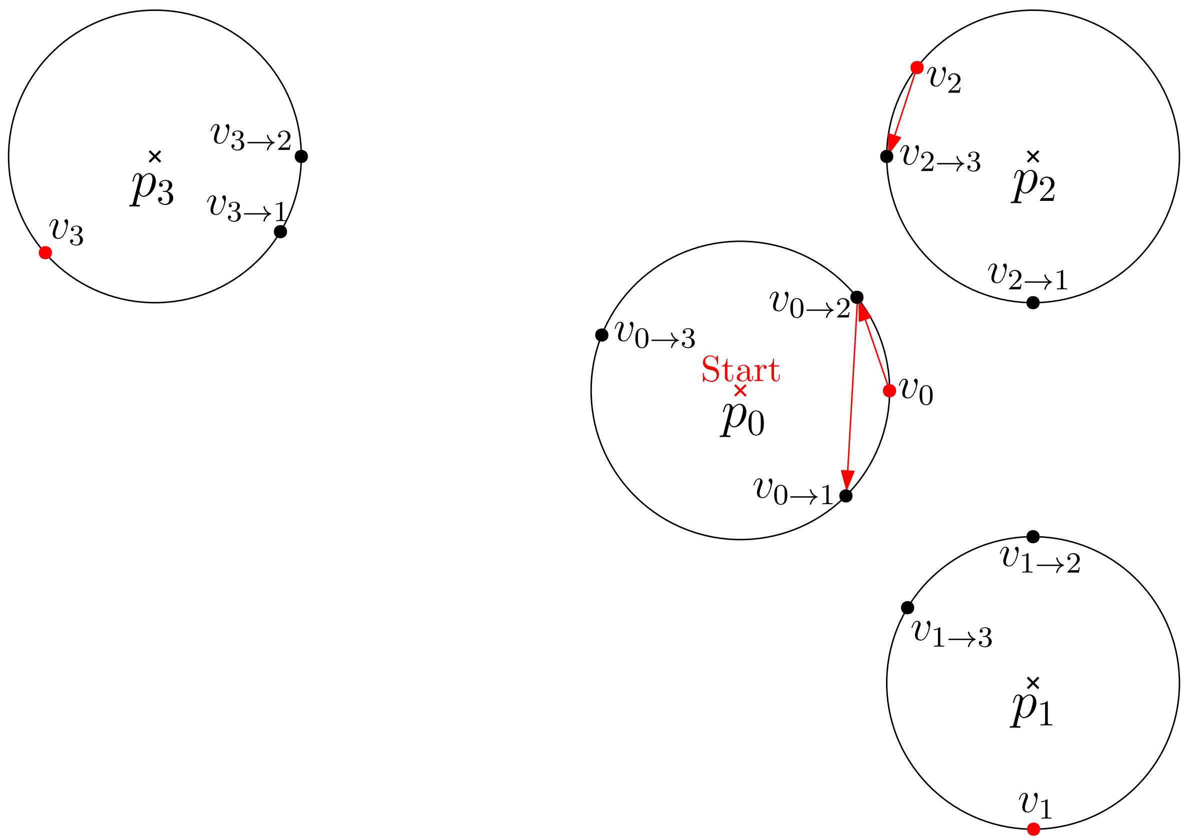
$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

$$y_w \geq y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \leq |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \leq \sum_{E_{\text{in}}(v_{i\to j})} x_e \leq 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \leq 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \leq |S| - 1 \qquad \forall S \subset V$$

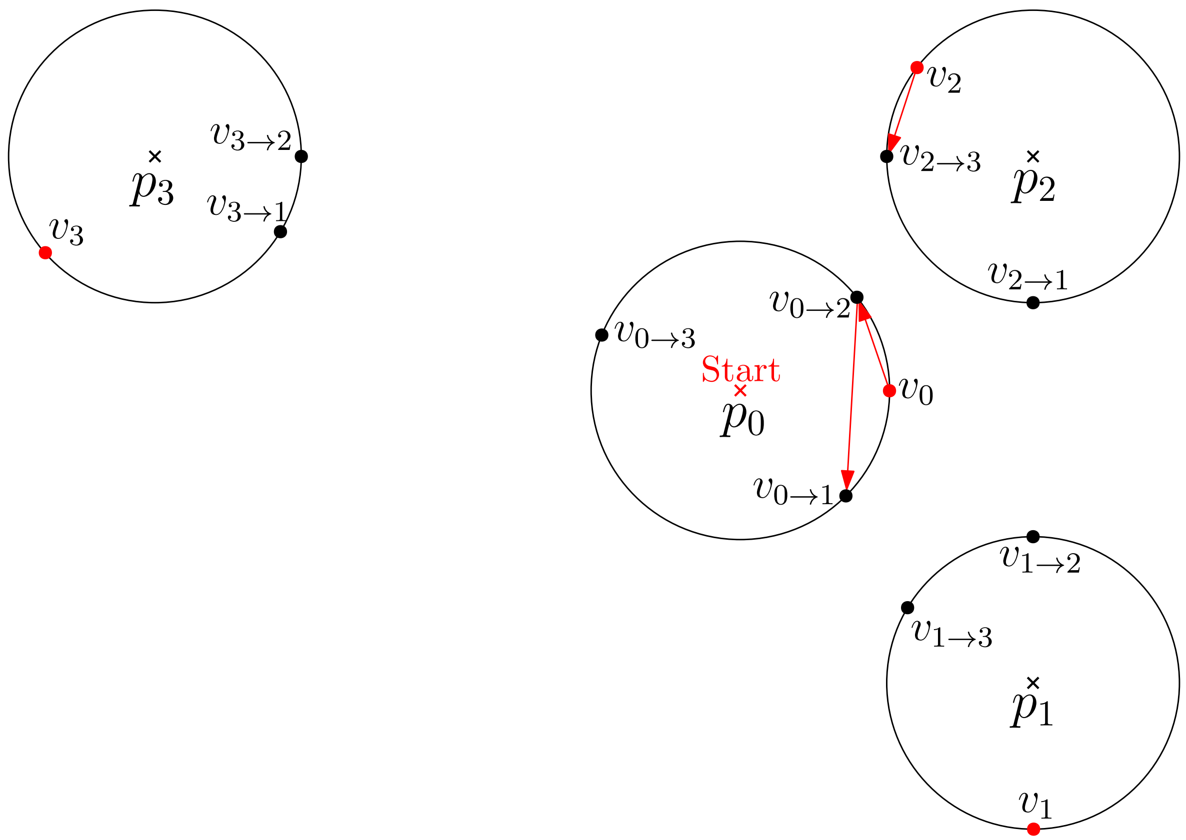
$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

$$y_w \geq y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \leq |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \leq \sum_{E_{\text{in}}(v_{i\to j})} x_e \leq 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \leq 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \leq |S| - 1 \qquad \forall S \subset V$$

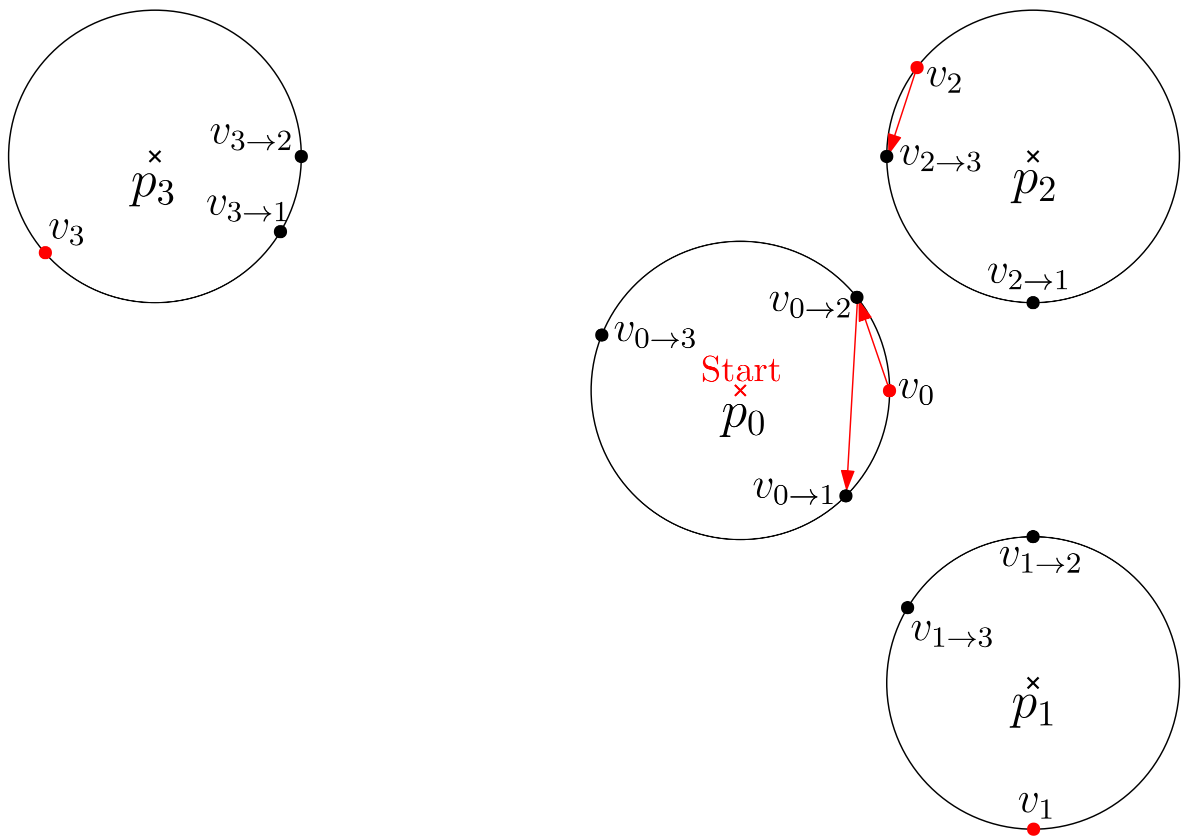
$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

$$y_w \geq y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \leq |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \leq \sum_{E_{\text{in}}(v_{i\to j})} x_e \leq 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \leq 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \leq |S| - 1 \qquad \forall S \subset V$$

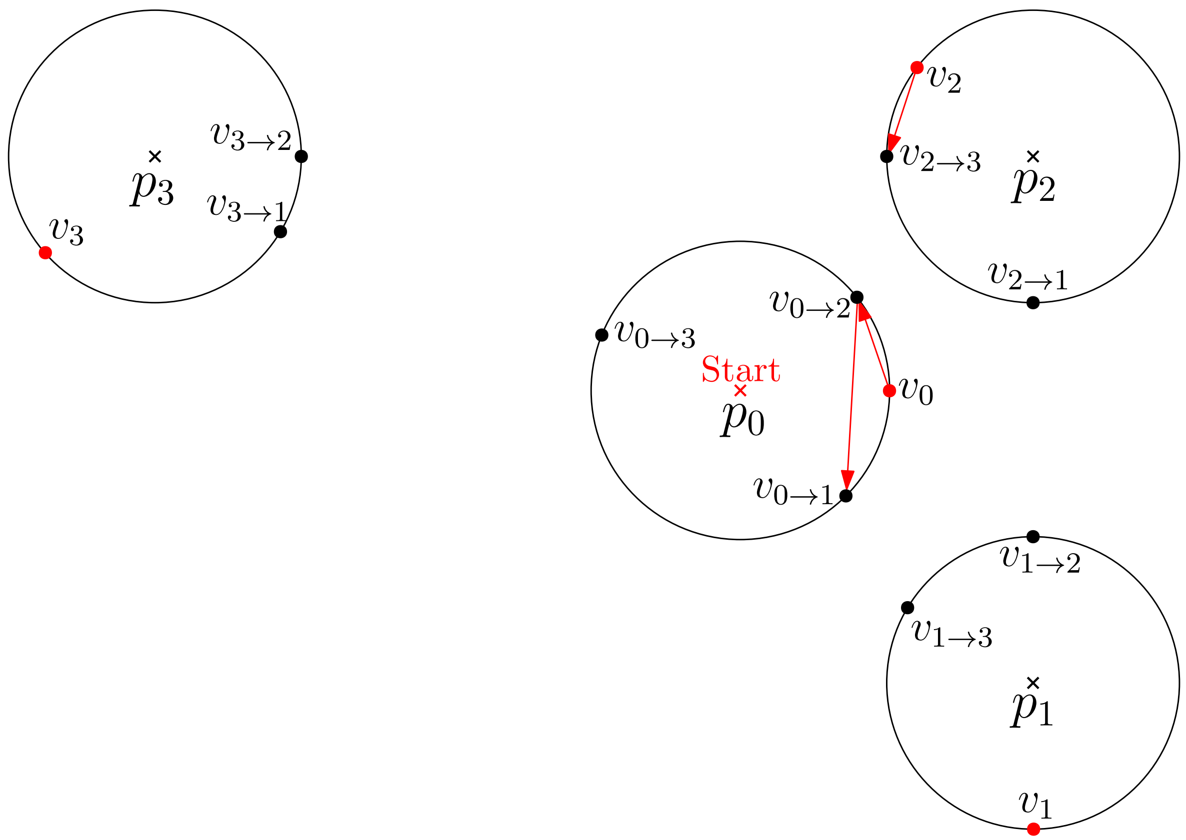
$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

$$y_w \geq y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \leq |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \le \sum_{E_{\text{in}}(v_{i\to j})} x_e \le 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \le 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \le |S| - 1 \qquad \forall S \subset V$$

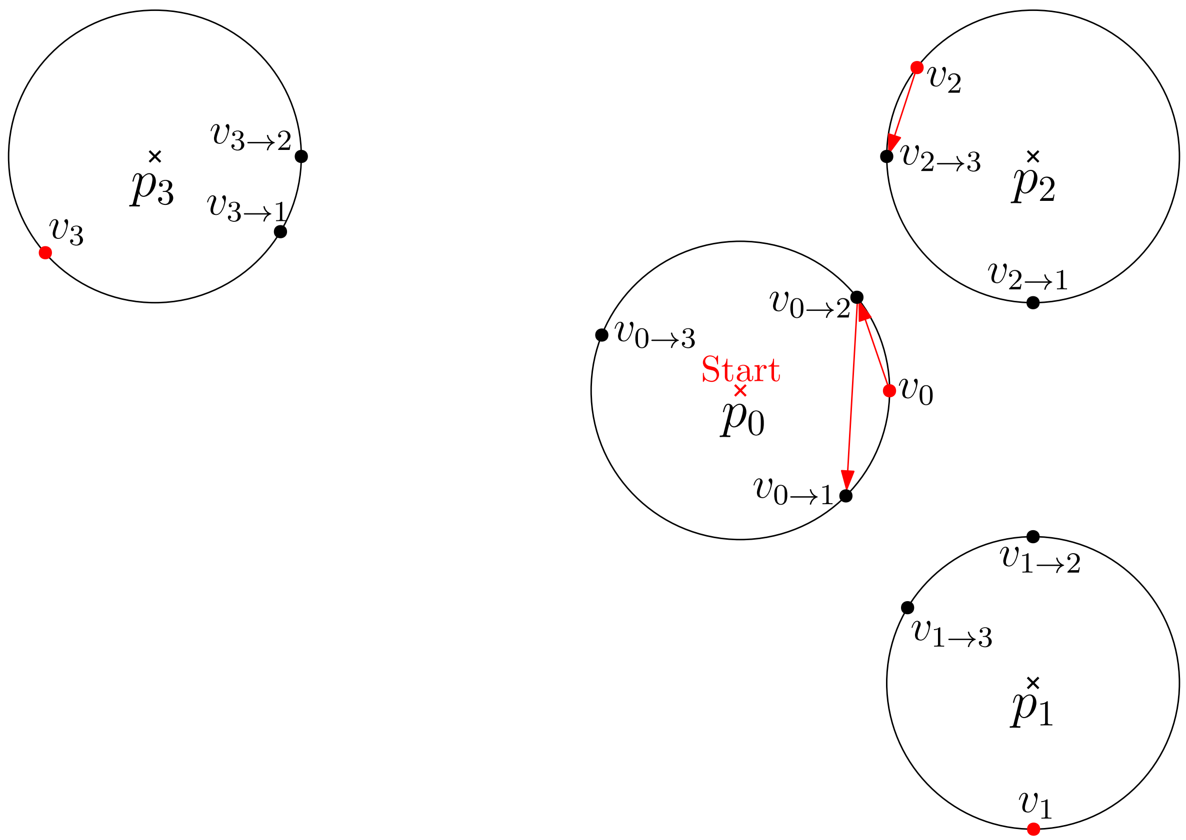
$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

$$y_w \ge y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \le |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \leq \sum_{E_{\text{in}}(v_{i\to j})} x_e \leq 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \leq 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \leq |S| - 1 \qquad \forall S \subset V$$

$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

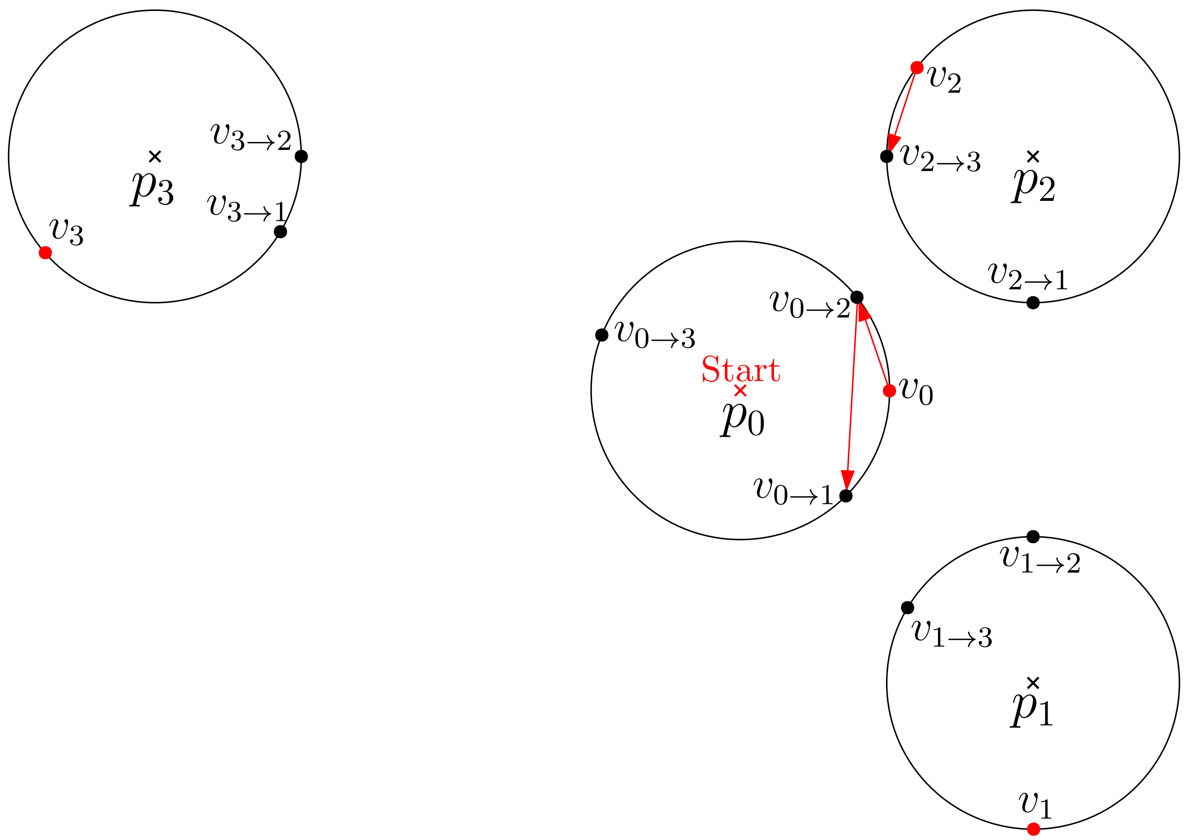
$$y_w \geq y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \leq |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de

Institute of Operating Systems and Computer Networks

Integer Programming



$$\min \max_{p_i \in P} y_{v_i}$$

$$\sum_{e \in E_{\text{in}}(v_{j\to i}), p_j \in P} x_e = 1 \qquad \forall p_i \in P \setminus \{p_0\}$$

$$\sum_{E_{\text{out}}(v_{i\to j})} x_e \le \sum_{E_{\text{in}}(v_{i\to j})} x_e \le 1 \qquad \forall v_{i\to j} \in V$$

$$\sum_{E_{\text{out}}(v_i)} x_e \le 1 \qquad \forall p_i \in P$$

$$\sum_{v,w \in S} x_{vw} \le |S| - 1 \qquad \forall S \subset V$$

$$y_{v_i} = \sum_{p_j \in P} y_{v_{j\to i}} \qquad \forall p_i \in P \setminus \{p_0\}$$

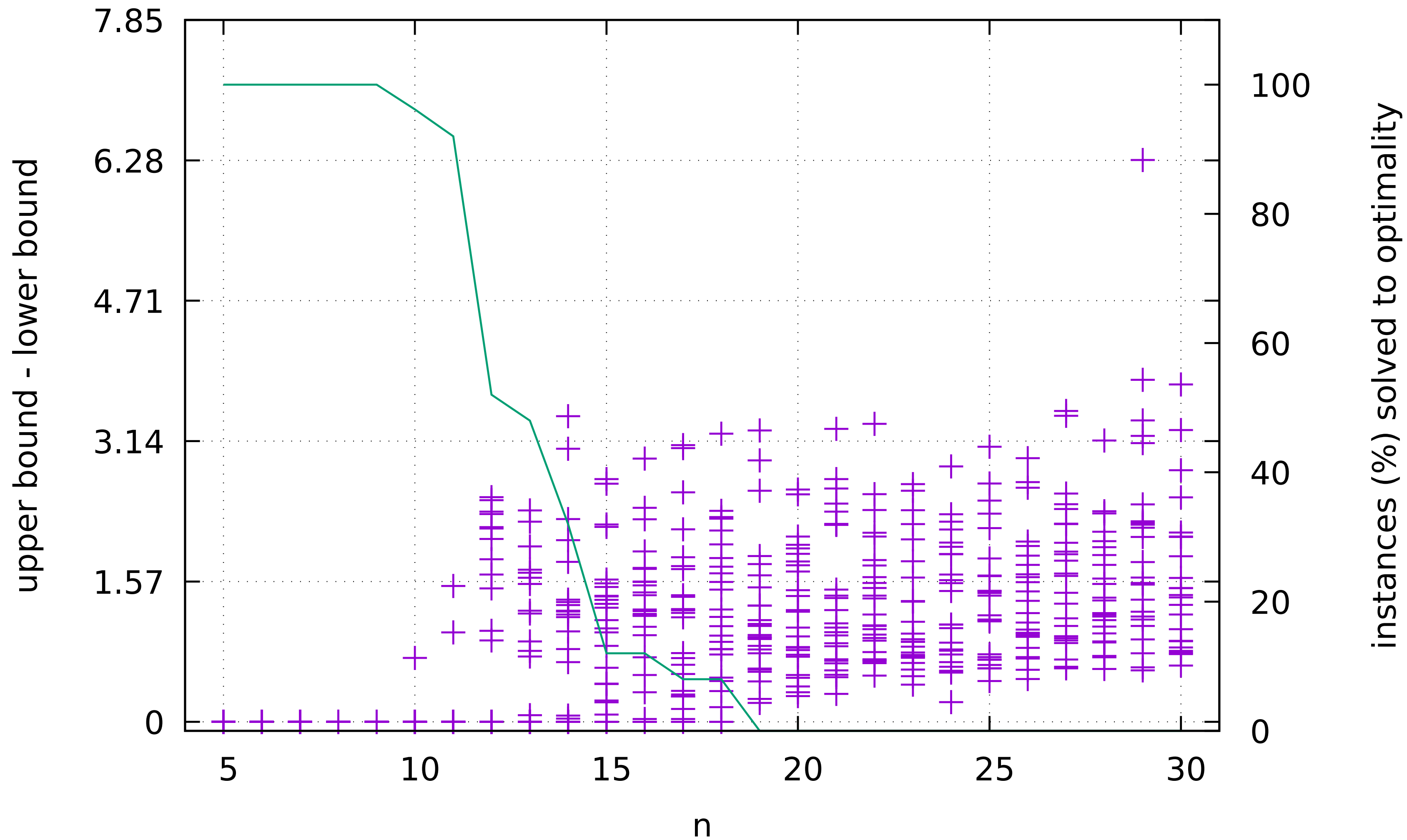
$$y_w \ge y_v + \text{cost}(vw) + (3\pi x_{vw} - 3\pi) \qquad \forall vw \in E$$

$$\sum_{p_i, p_j \in S} \sum_{e \in E_{\text{in}}(v_{i\to j})} x_e \le |S| - 1 \qquad \forall S \subset P \setminus \{p_0\}$$

d.krupke@tu-bs.de

Institute of Operating Systems and Computer Networks

Experiments



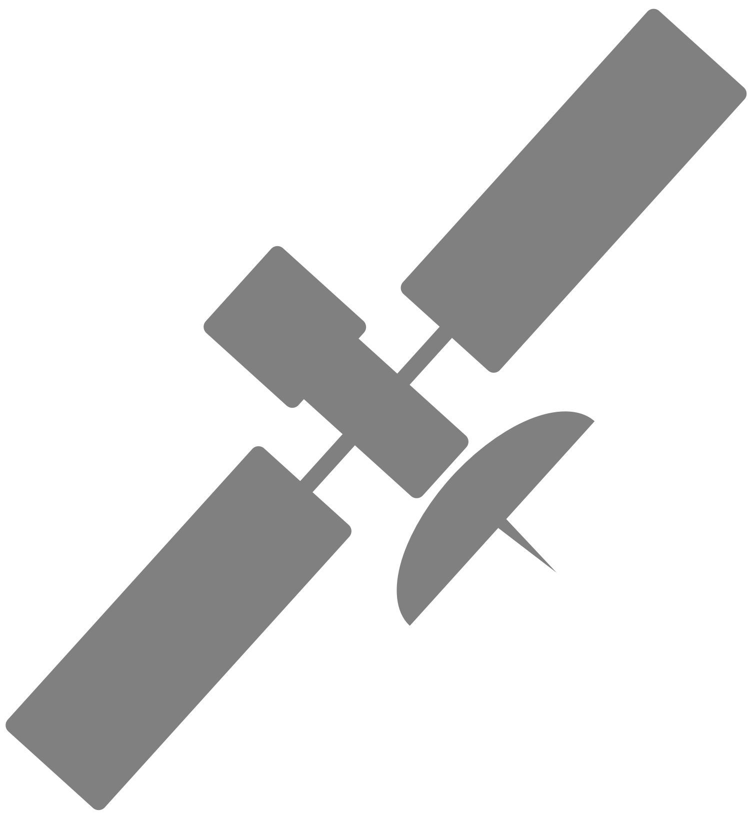
Technische Universität Braunschweig
d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

Conclusion

d.krupke@tu-bs.de

Institute of Operating Systems and Computer Networks

This is just the beginning. . .

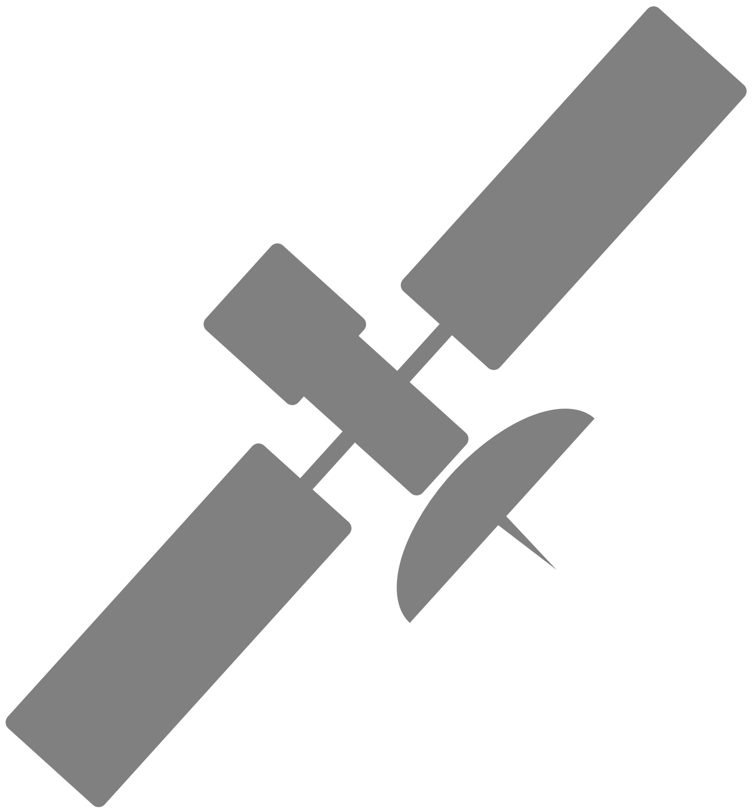


Still to do:

- Adjustment of receiver
- Transmission time and delay
- Satellites can prepare before they actually receive the data
- 3D, Sphere, Orbital Movement.
- Cost of changing rotation.

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

This is just the beginning...



Still to do:

- Adjustment of receiver
- Transmission time and delay
- Satellites can prepare before they actually receive the data
- 3D, Sphere, Orbital Movement.
- Cost of changing rotation.

Thank you for your attention!

d.krupke@tu-bs.de
Institute of Operating Systems and Computer Networks

1. Introduction
2. Freeze Tag
3. Angular Freeze Tag
4. **Angular Scan Cover**

Angular Scan Cover

Angular Scan Cover

SIAM J. DISCRETE MATH.
Vol. 35, No. 2, pp. 1337–1355

MINIMUM SCAN COVER WITH ANGULAR TRANSITION COSTS*

SÁNDOR P. FEKETE†, LINDA KLEIST†, AND DOMINIK KRUPKE†

Abstract. We provide a comprehensive study of a natural graph optimization problem that arises from transition costs between incident edges. In the problem MINIMUM SCAN COVER WITH ANGULAR COSTS (MSC), we are given a graph G that is embedded in Euclidean space. The edges of G need to be *scanned*, i.e., probed from both of their vertices. In order to scan their edge, two vertices need to face each other; changing the heading of a vertex takes some time proportional to the corresponding turn angle. Our goal is to minimize the time until all scans are completed, i.e., to compute a schedule of minimum makespan. A real-world motivation arises in the context of satellite communication and astrophysics. We show that MSC is closely related to both graph coloring and the minimum (directed and undirected) cut cover problem; in particular, we show that the minimum scan time for instances in 1D and 2D lies in $\Theta(\log \chi(G))$, while for 3D the minimum scan time is not upper bounded by $\chi(G)$. We use this relationship to prove that the existence of a constant-factor approximation implies P = NP, even for one-dimensional instances. In 2D, we show that it is NP-hard to approximate a minimum scan cover within less than a factor of $^3/_2$, even for bipartite graphs; conversely, we present a 9/2-approximation algorithm for this scenario. Generally, we give an $O(c)$-approximation for k-colored graphs with $k \leq \chi(G)^c$. For general metric cost functions, we provide approximation algorithms whose performance guarantees depend on the arboricity of the graph.

Key words. graph scanning, graph coloring, angular metric, approximation, scheduling

AMS subject classifications. 05C10, 05C15, 51F99, 52C45, 68U05, 90B35, 90C27

DOI. 10.1137/20M1368161

1. Introduction. Many problems of graph optimization arise from questions of communication, where different locations need to be connected. In many scenarios, this implies objective functions in which the cost of a connection is based on the geometric *distance* between the involved vertices; a practical example arises in wire-based transmission, where the length of an electro-optic link corresponds to connection

Technische Universität Braunschweig

Technische Universität Braunschweig

Student Presentation

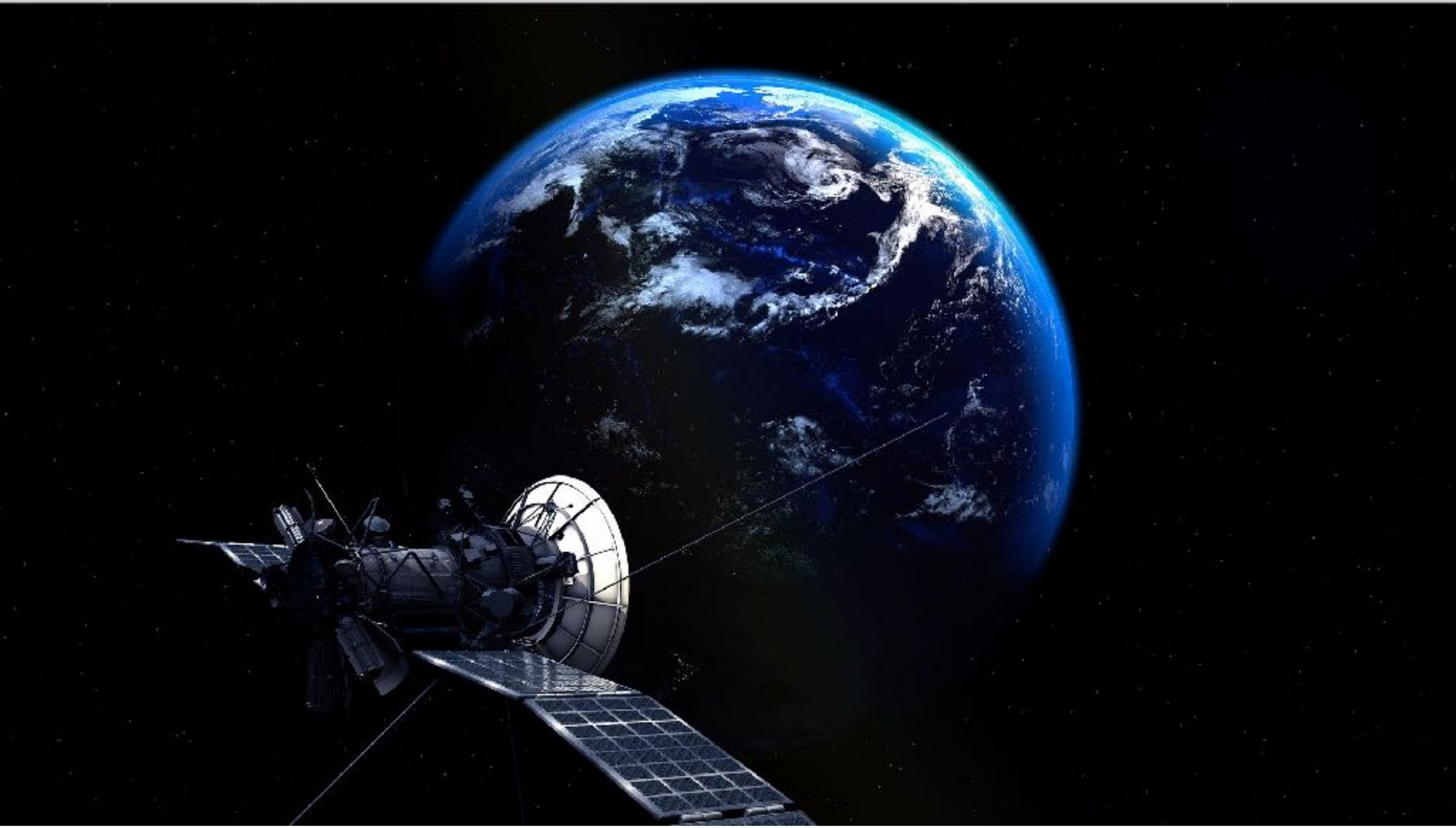
Minimum Scan Cover with Angular Transition Costs

Sándor Fekete

Linda Kleist

Dominik Krupke

Motivation: Directed Antennas



Motivation: Directed Antennas

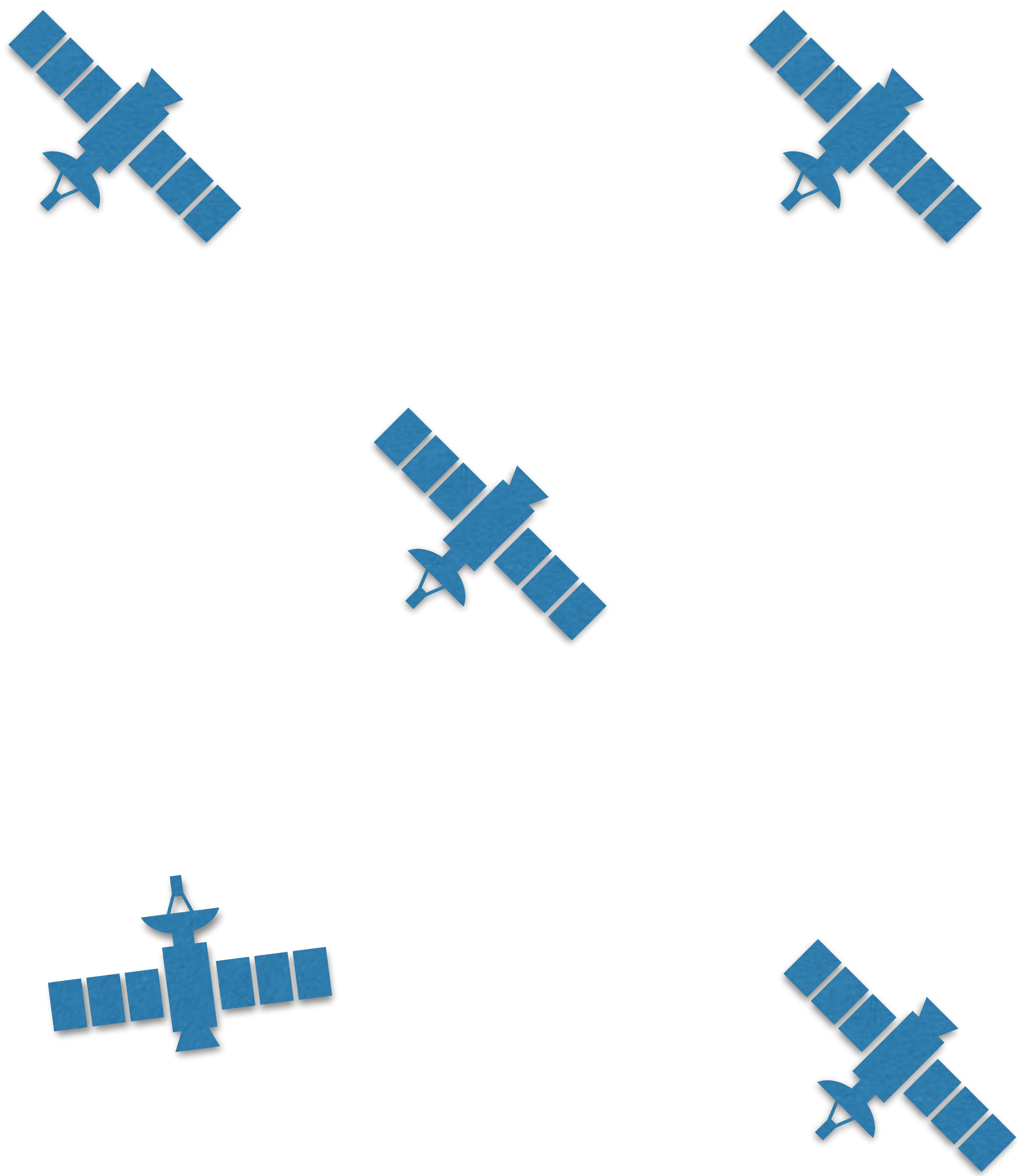
Technische Universität Braunschweig
d.krupke@tu-bs.de

Motivation: Directed Antennas

Motivation: Directed Antennas

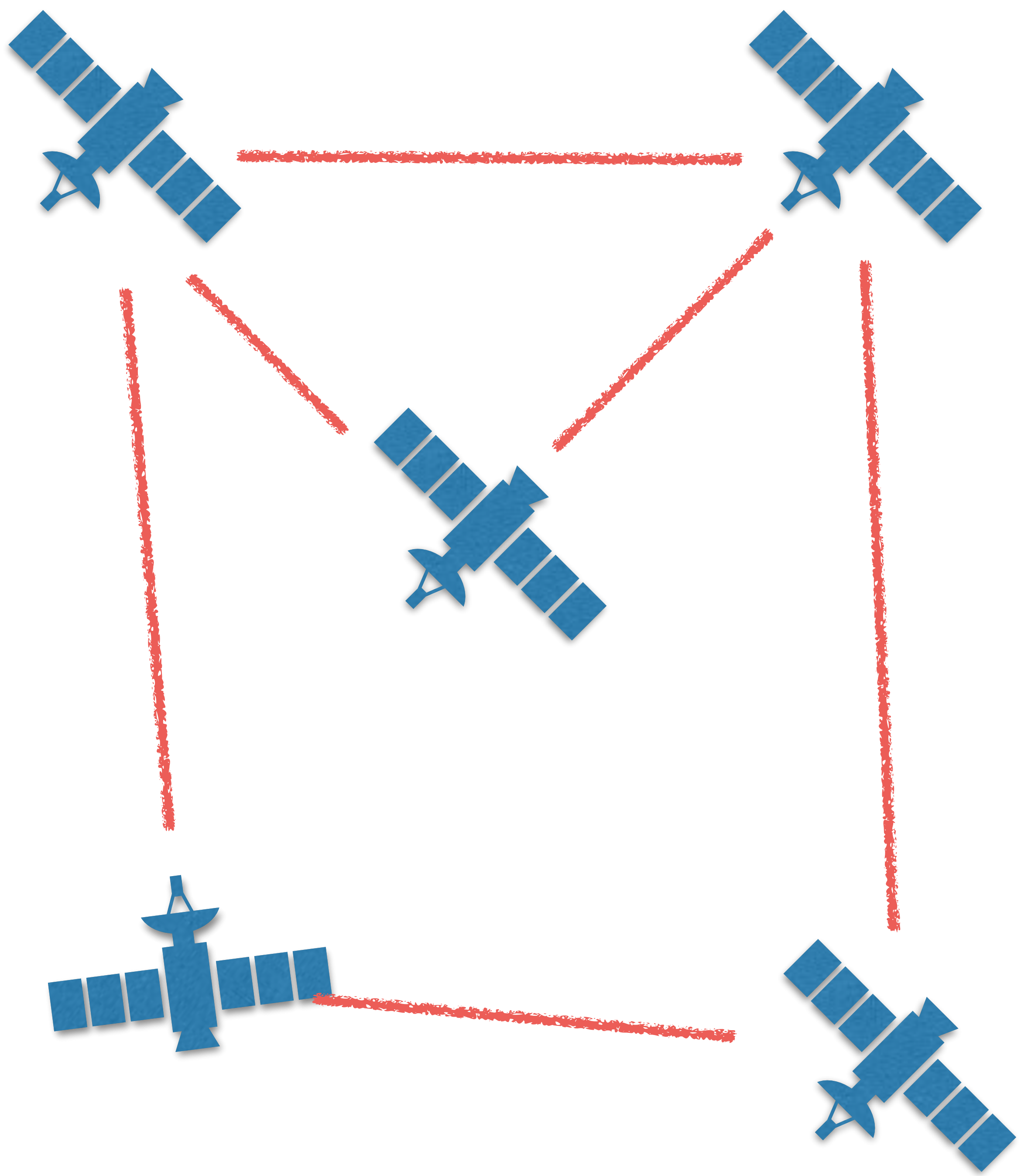
Multiple Tasks

Multiple Tasks



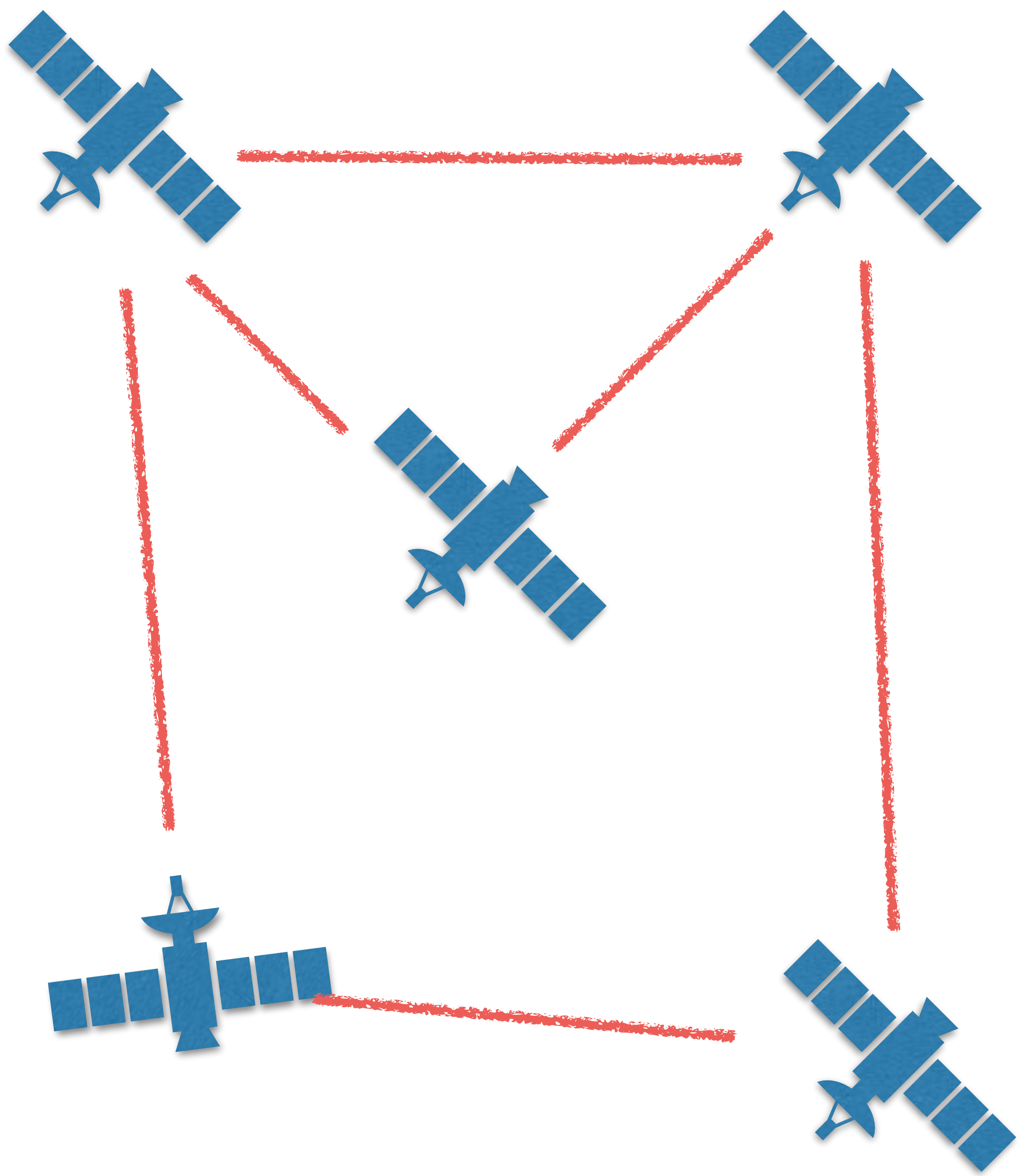
Technische Universität Braunschweig
d.krupke@tu-bs.de

Multiple Tasks



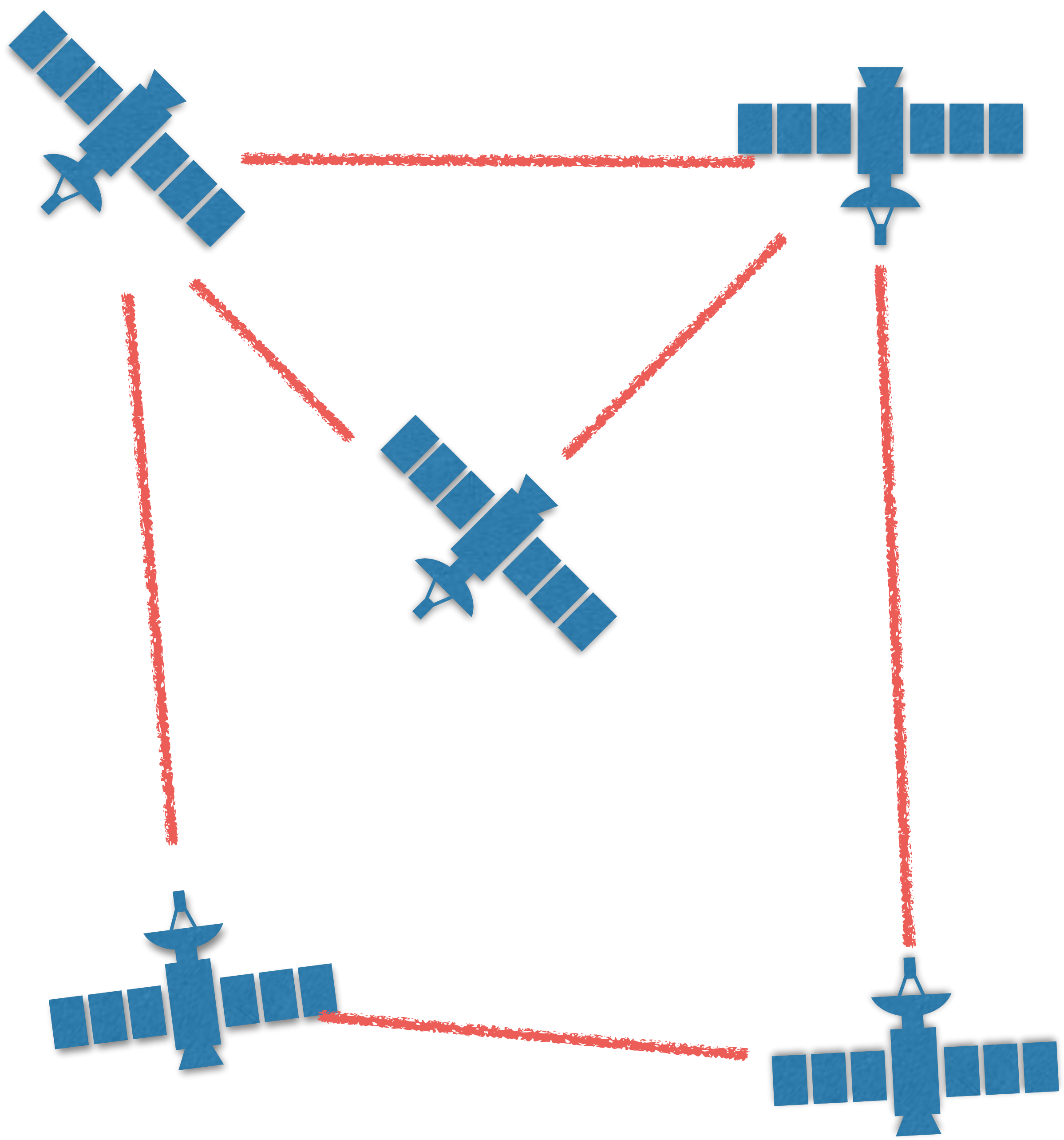
Technische Universität Braunschweig
d.krupke@tu-bs.de

Multiple Tasks



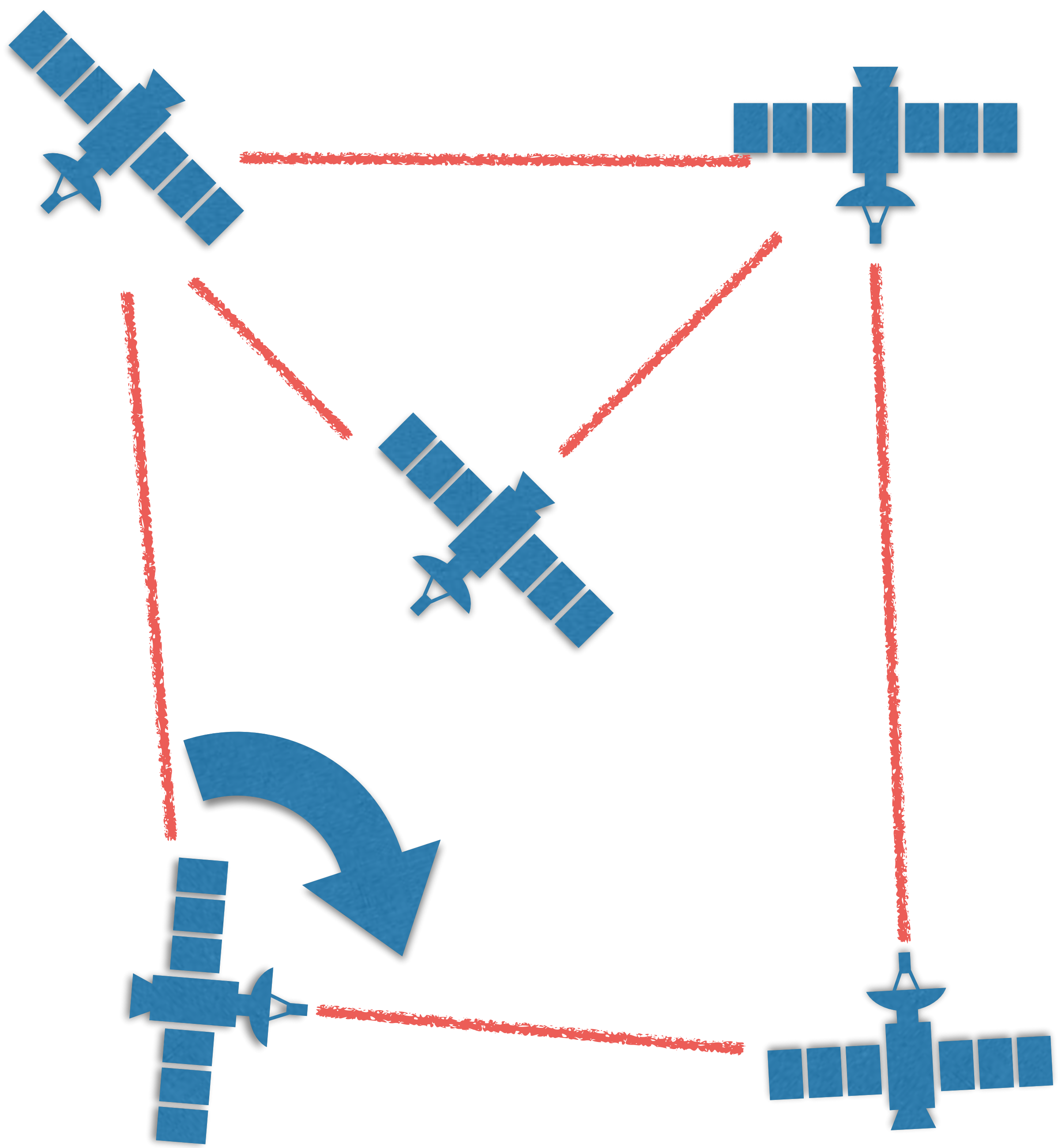
- 'Scan' all edges

Multiple Tasks



- 'Scan' all edges
- Immediate scan if both satellites point to each other

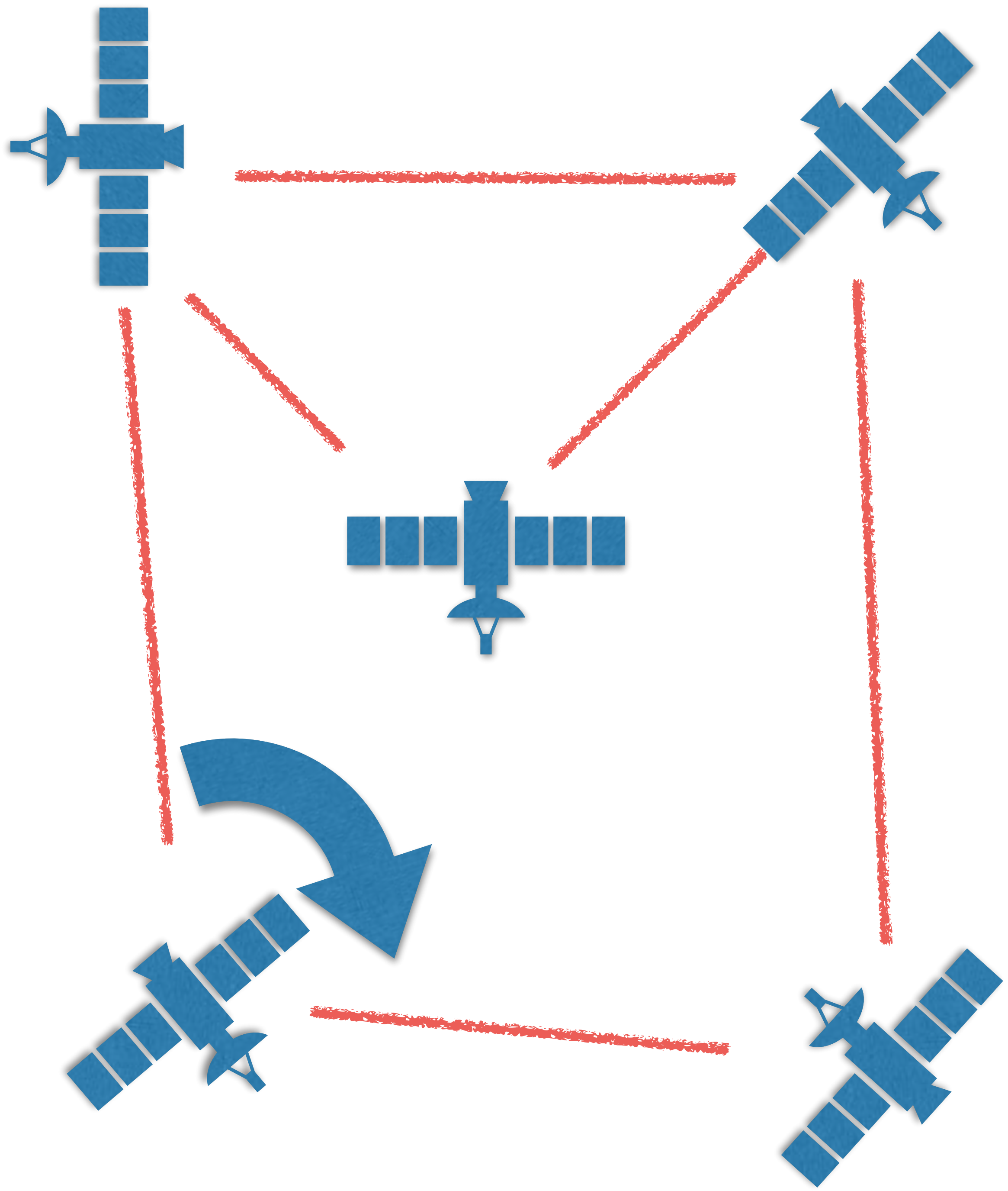
Multiple Tasks



- 'Scan' all edges
- Immediate scan if both satellites point to each other
- Rotation time = angle

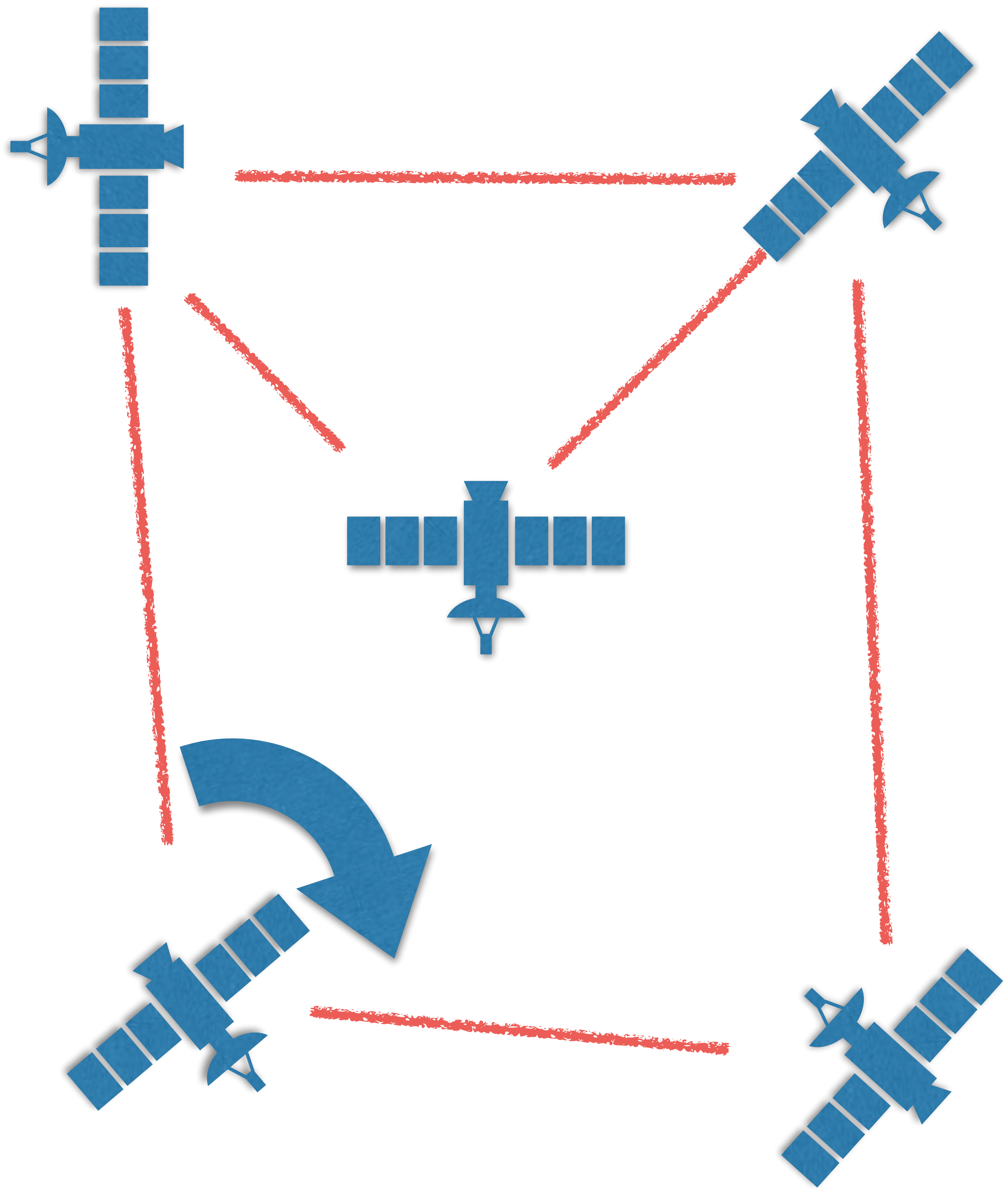
d.krupke@tu-bs.de

Multiple Tasks



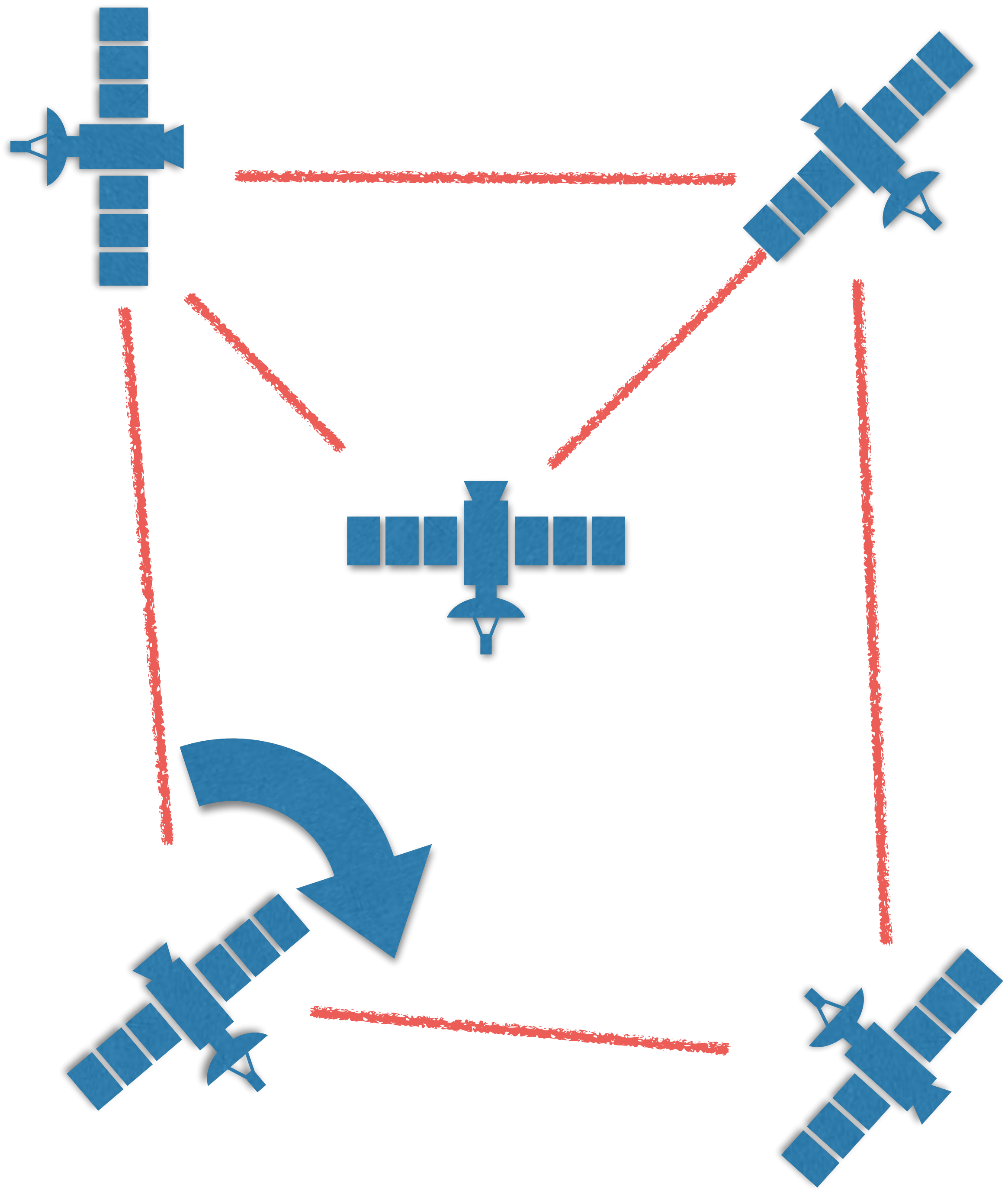
- 'Scan' all edges
- Immediate scan if both satellites point to each other
- Rotation time = angle
- Can rotate in parallel

Multiple Tasks



- 'Scan' all edges
- Immediate scan if both satellites point to each other
- Rotation time = angle
- Can rotate in parallel
- Start heading can be chosen

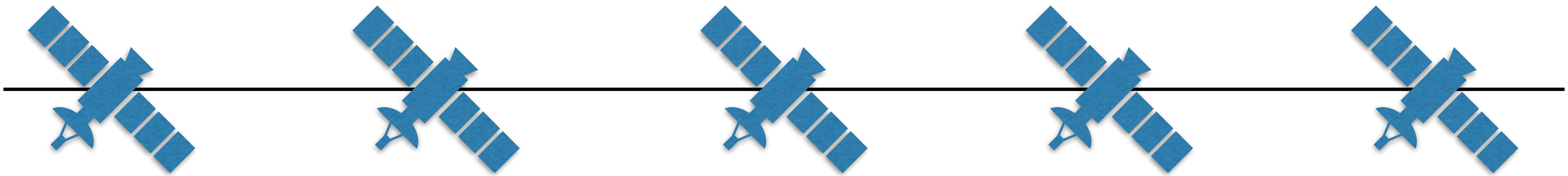
Multiple Tasks



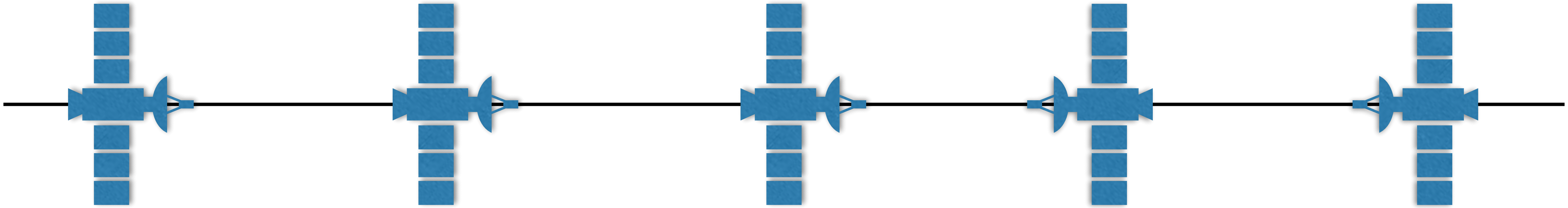
- 'Scan' all edges
- Immediate scan if both satellites point to each other
- Rotation time = angle
- Can rotate in parallel
- Start heading can be chosen

Minimize makespan

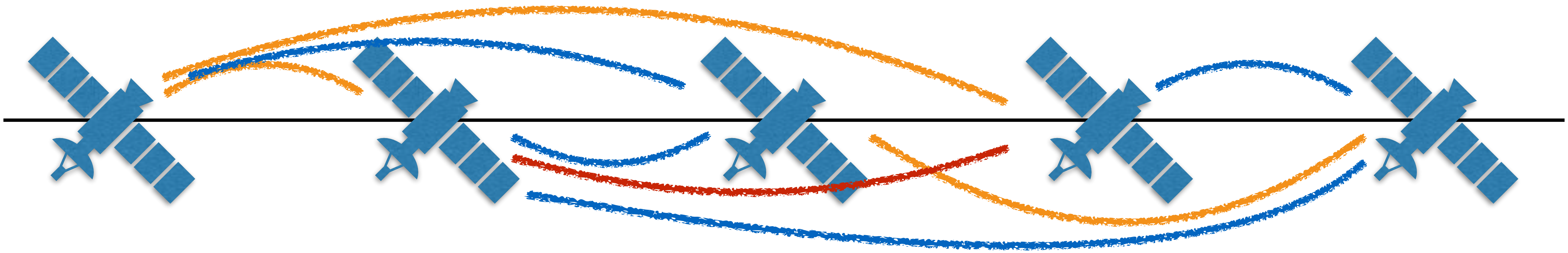
1-D



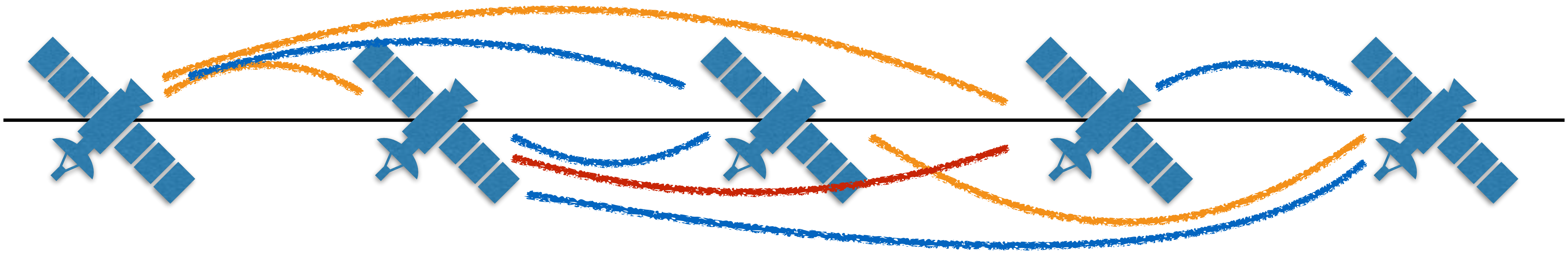
1-D



1-Dimensional

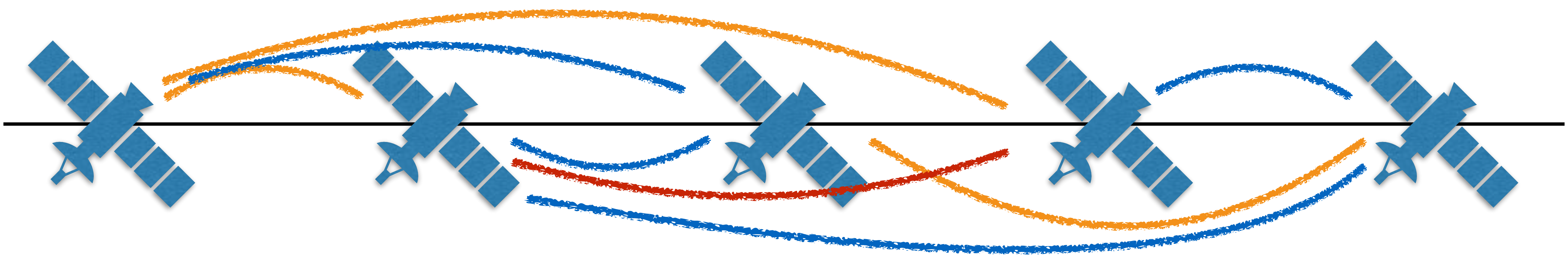


1-Dimensional



Discrete 180° steps!

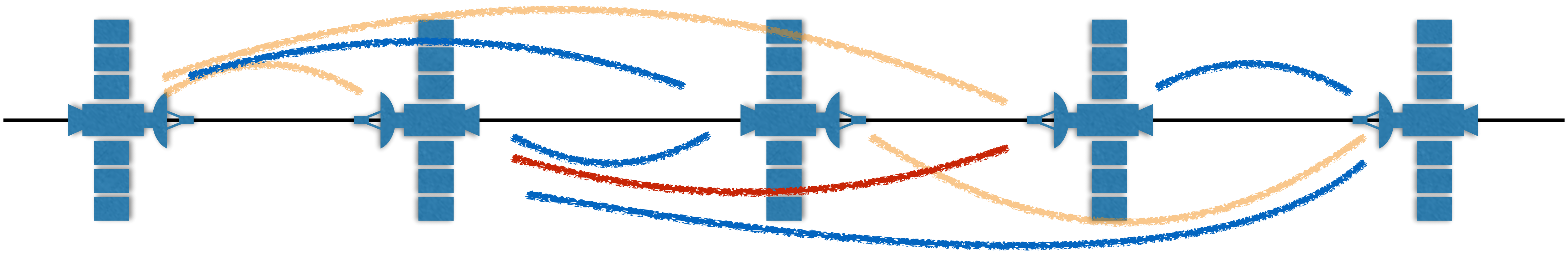
1-Dimensional



Discrete $180°$ steps!

$$\begin{pmatrix}\rightarrow\\\rightarrow\\\rightarrow\end{pmatrix} \quad \begin{pmatrix}\leftarrow\\\rightarrow\\\rightarrow\end{pmatrix} \quad \begin{pmatrix}\rightarrow\\\leftarrow\\\leftarrow\end{pmatrix} \quad \begin{pmatrix}\leftarrow\\\rightarrow\\\leftarrow\end{pmatrix} \quad \begin{pmatrix}\leftarrow\\\leftarrow\\\leftarrow\end{pmatrix}$$

1-Dimensional

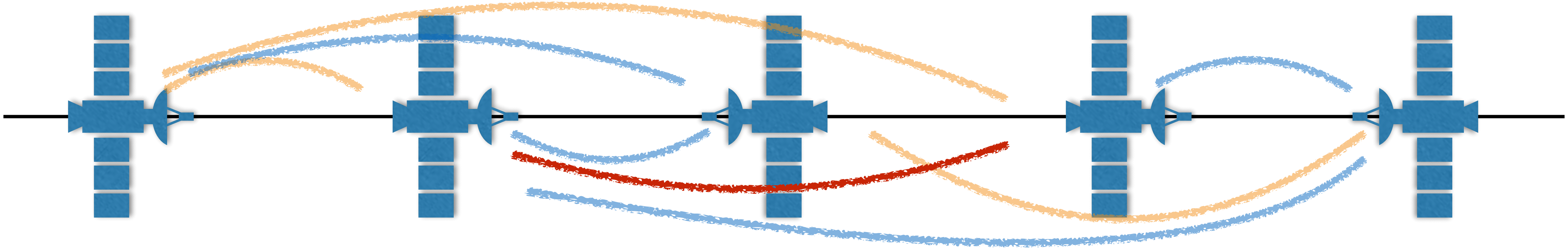


Discrete $180°$ steps!

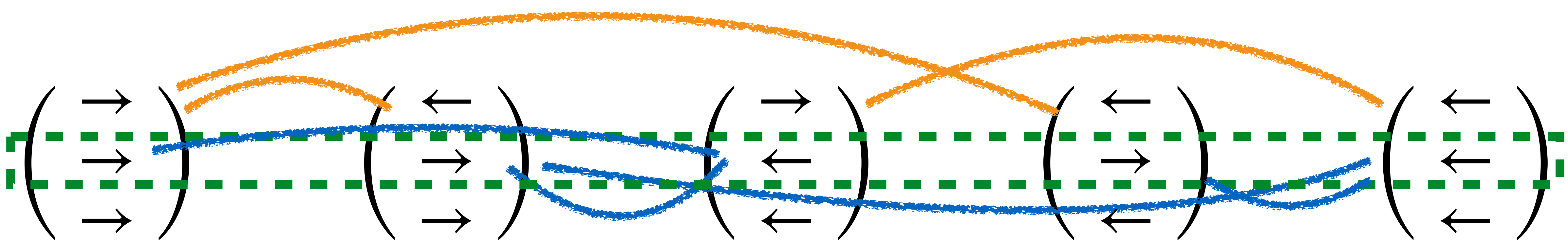
$$\begin{pmatrix}\rightarrow\\ \rightarrow\\ \rightarrow\end{pmatrix} \quad \begin{pmatrix}\leftarrow\\ \rightarrow\\ \rightarrow\end{pmatrix} \quad \begin{pmatrix}\rightarrow\\ \leftarrow\\ \leftarrow\end{pmatrix} \quad \begin{pmatrix}\leftarrow\\ \rightarrow\\ \leftarrow\end{pmatrix} \quad \begin{pmatrix}\leftarrow\\ \leftarrow\\ \leftarrow\end{pmatrix}$$

d.krupke@tu-bs.de

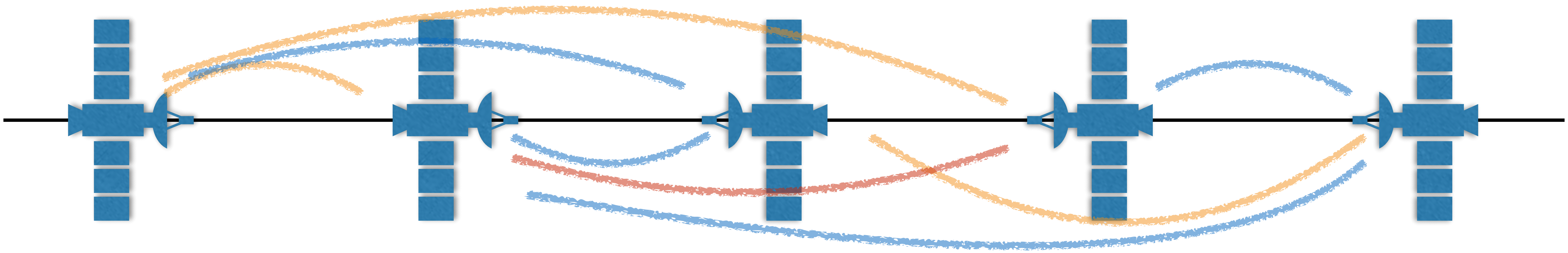
1-Dimensional



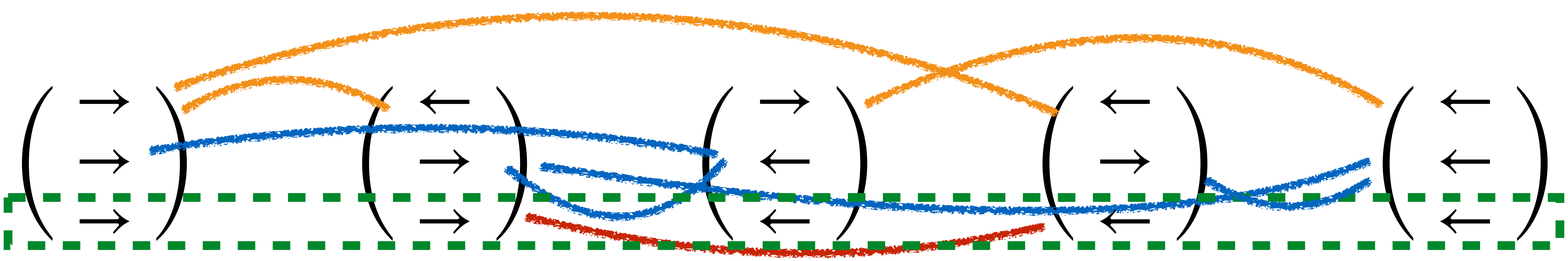
Discrete $180°$ steps!



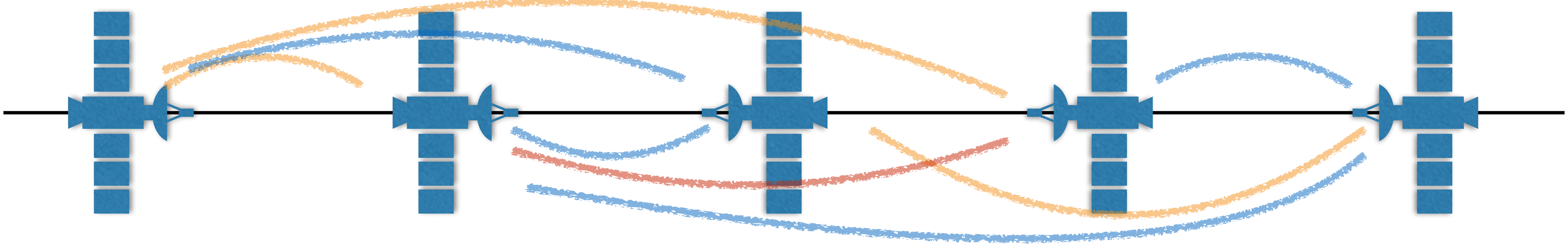
1-Dimensional



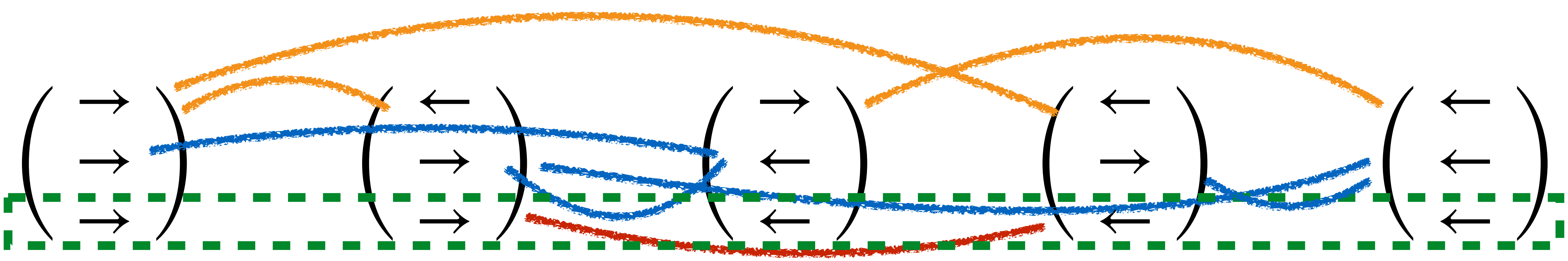
Discrete 180° steps!



1-Dimensional

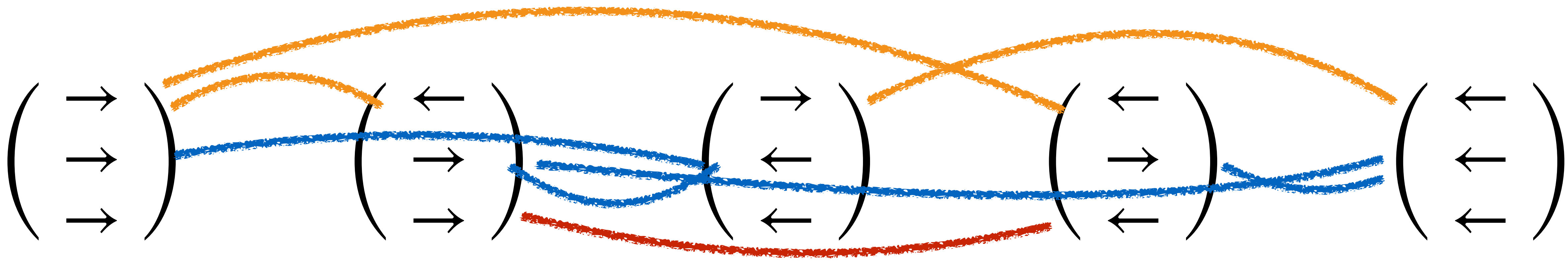


Discrete 180° steps!

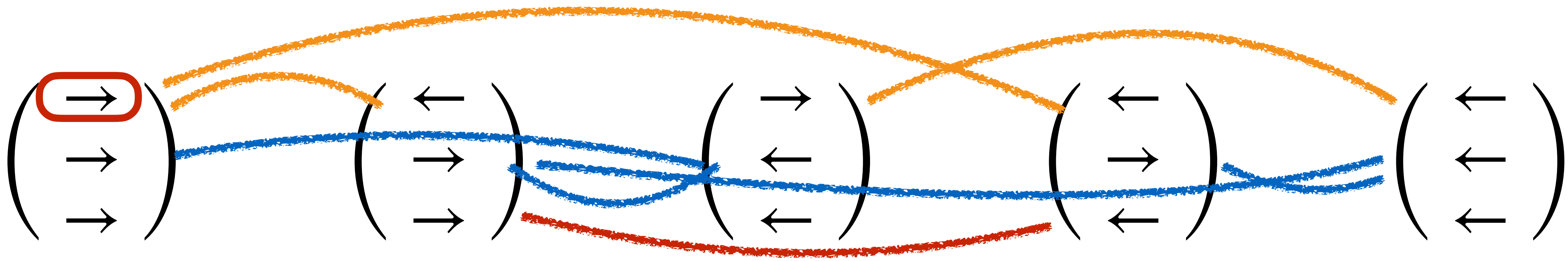


$2 \times 180^\circ = 360^\circ$ rotation time

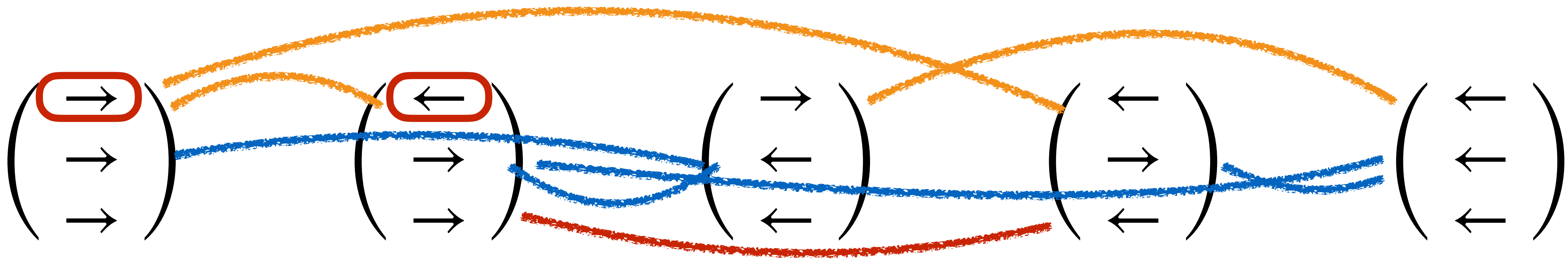
Bipartite Graphs



Bipartite Graphs

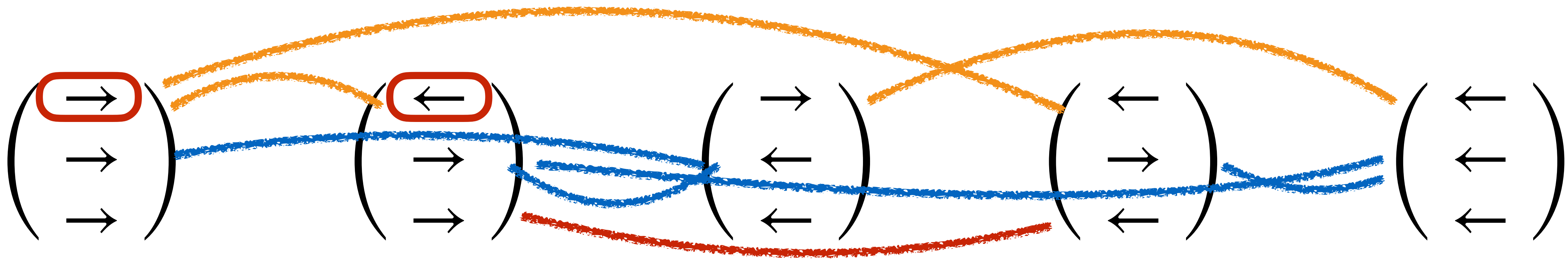


Bipartite Graphs



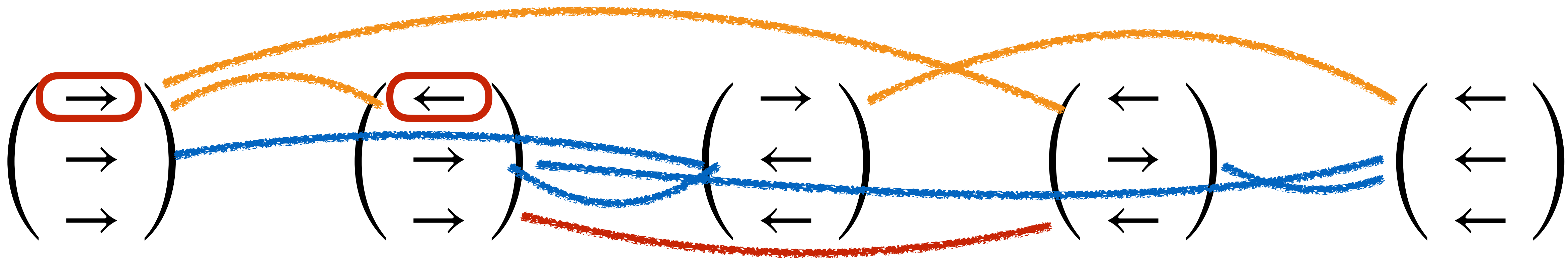
d.krupke@tu-bs.de

Bipartite Graphs



Every scan step is a bipartite graph!

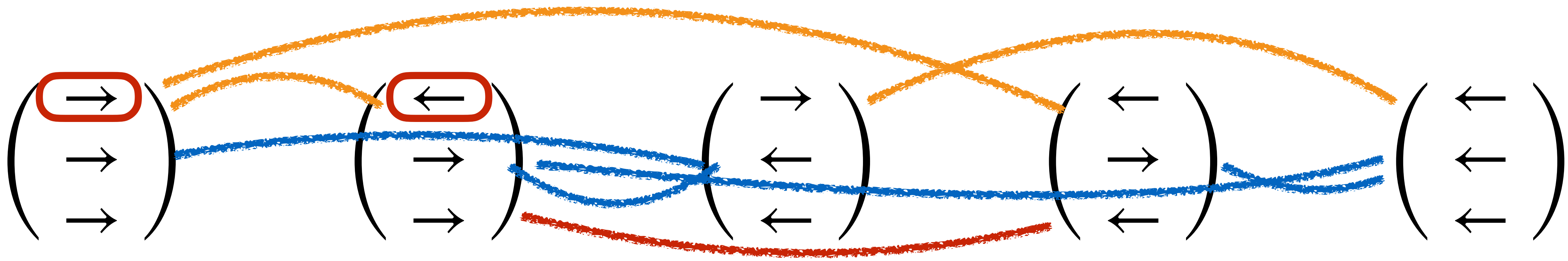
Bipartite Graphs



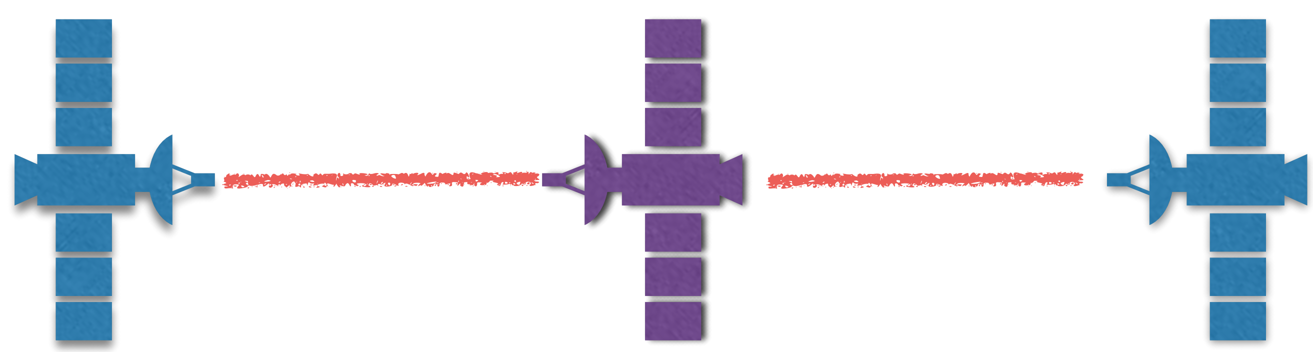
Every scan step is a bipartite graph!

d.krupke@tu-bs.de

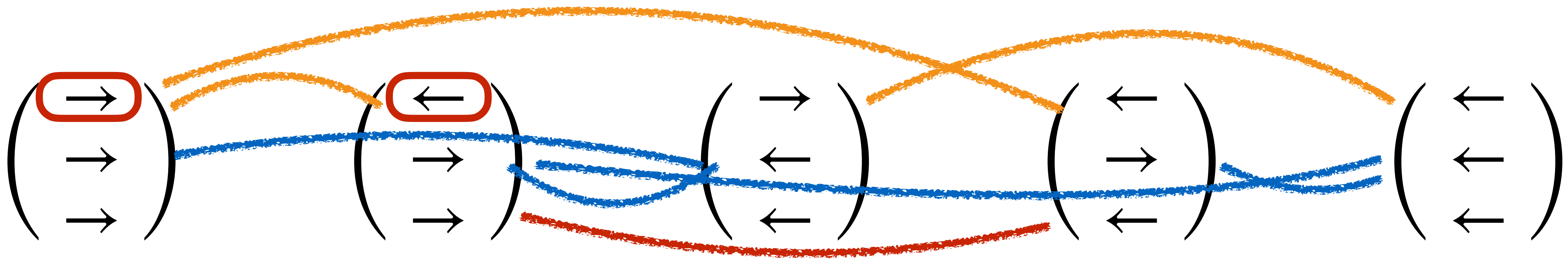
Bipartite Graphs



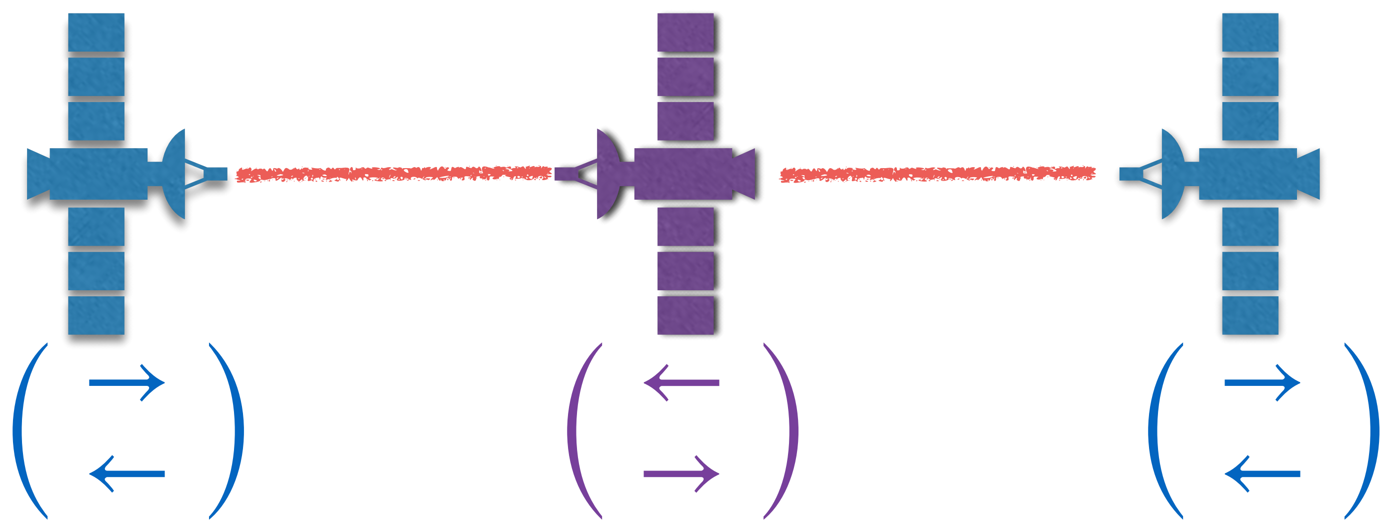
Every scan step is a bipartite graph!



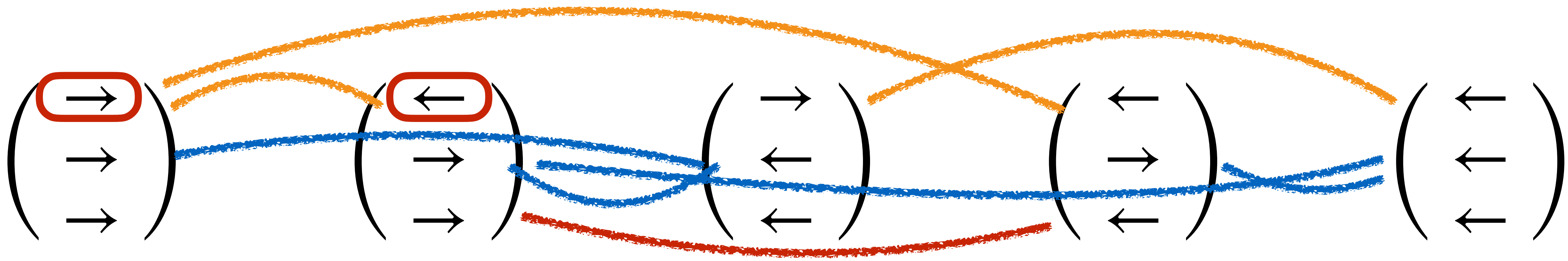
Bipartite Graphs



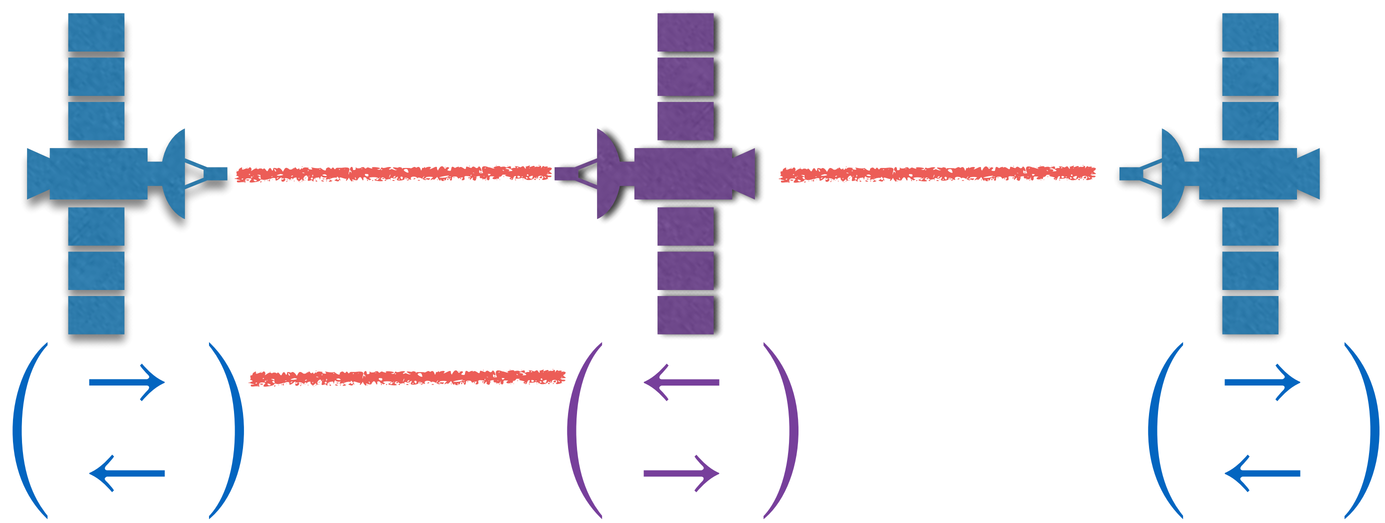
Every scan step is a bipartite graph!



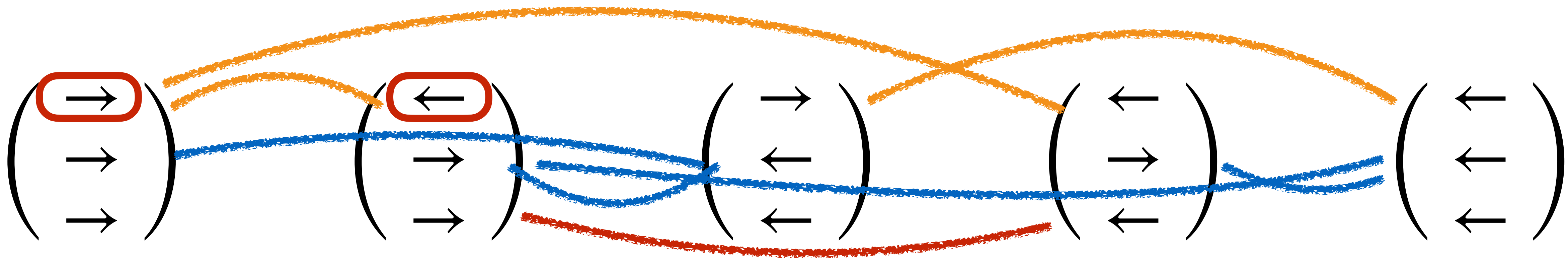
Bipartite Graphs



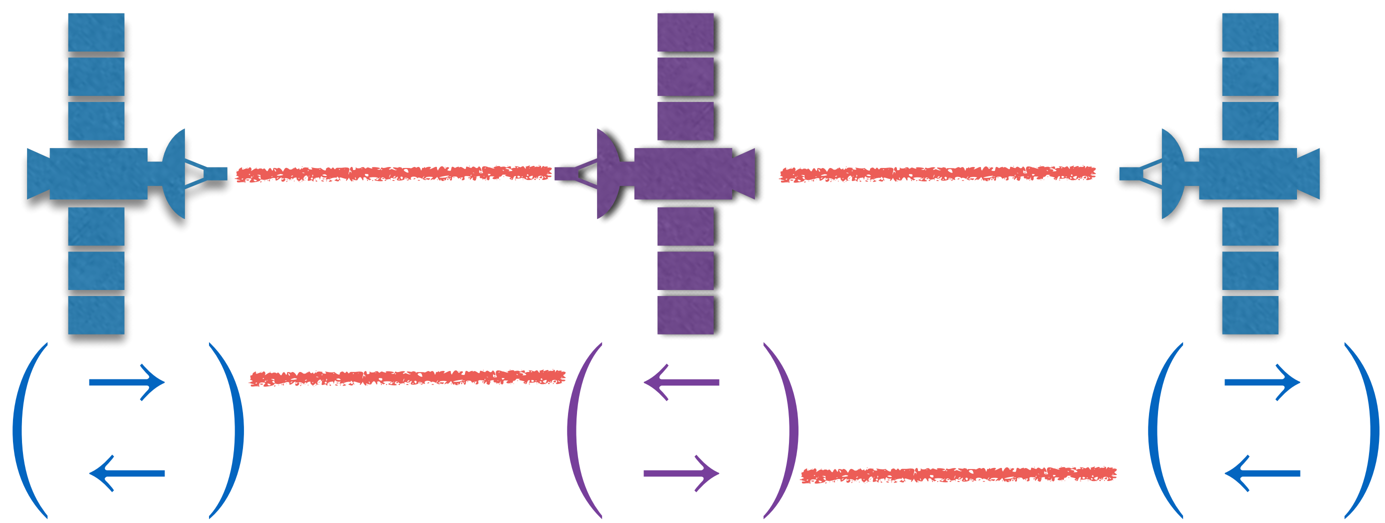
Every scan step is a bipartite graph!



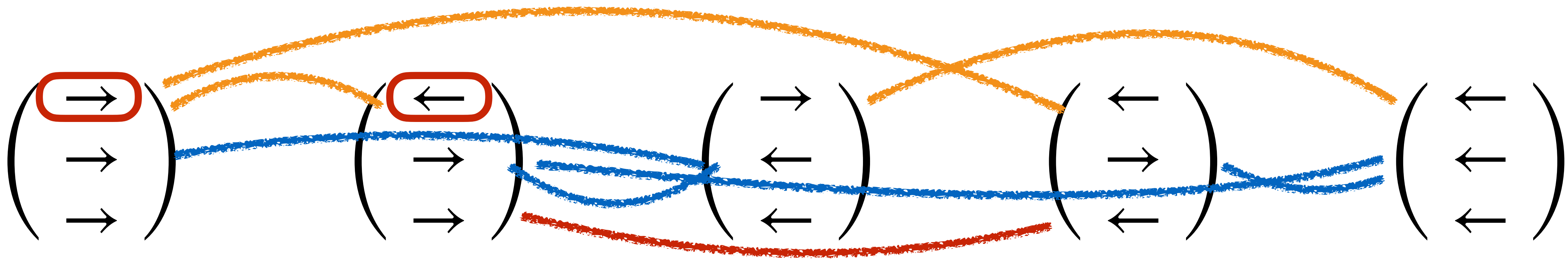
Bipartite Graphs



Every scan step is a bipartite graph!



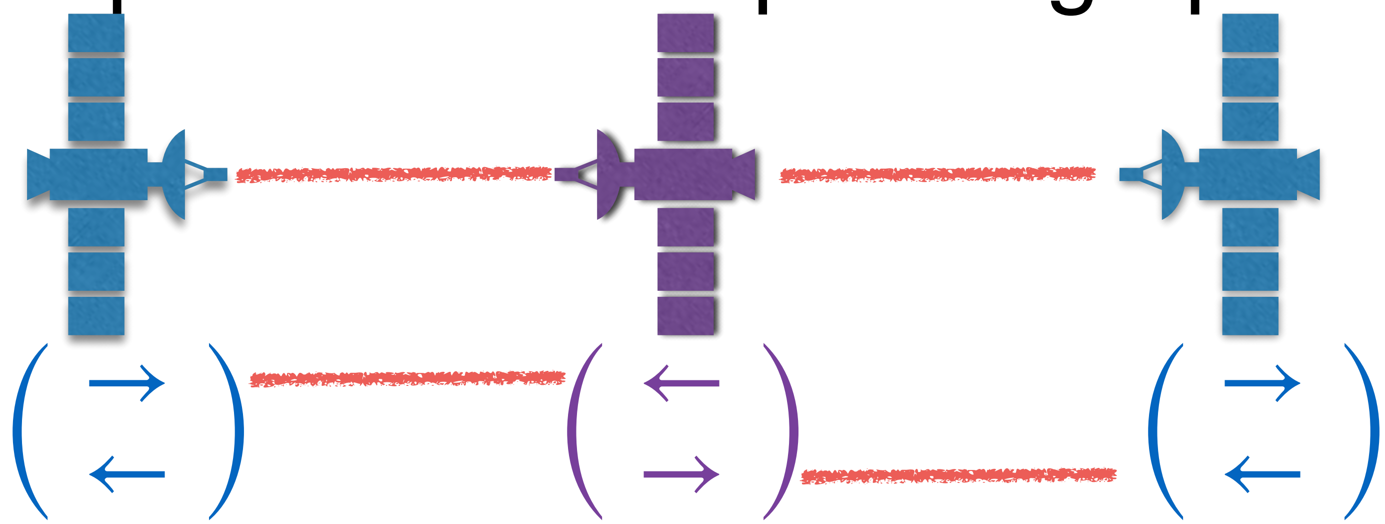
Bipartite Graphs



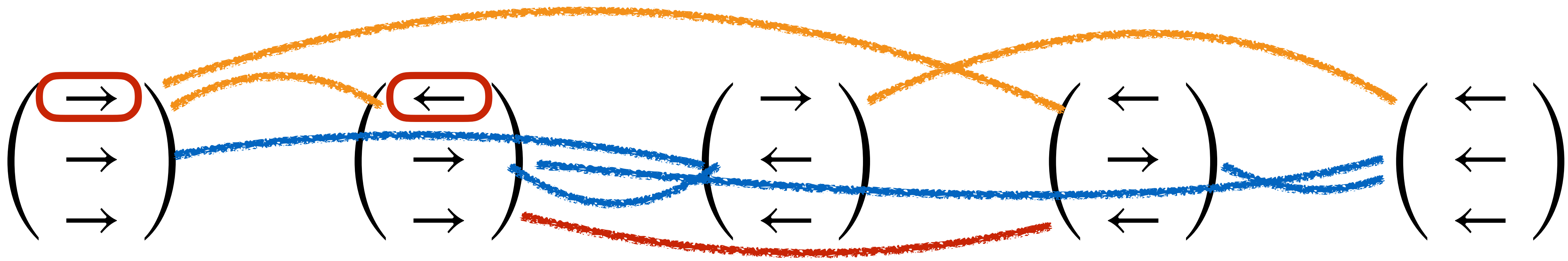
Every scan step is a bipartite graph!

---

2 steps suffice for bipartite graphs!



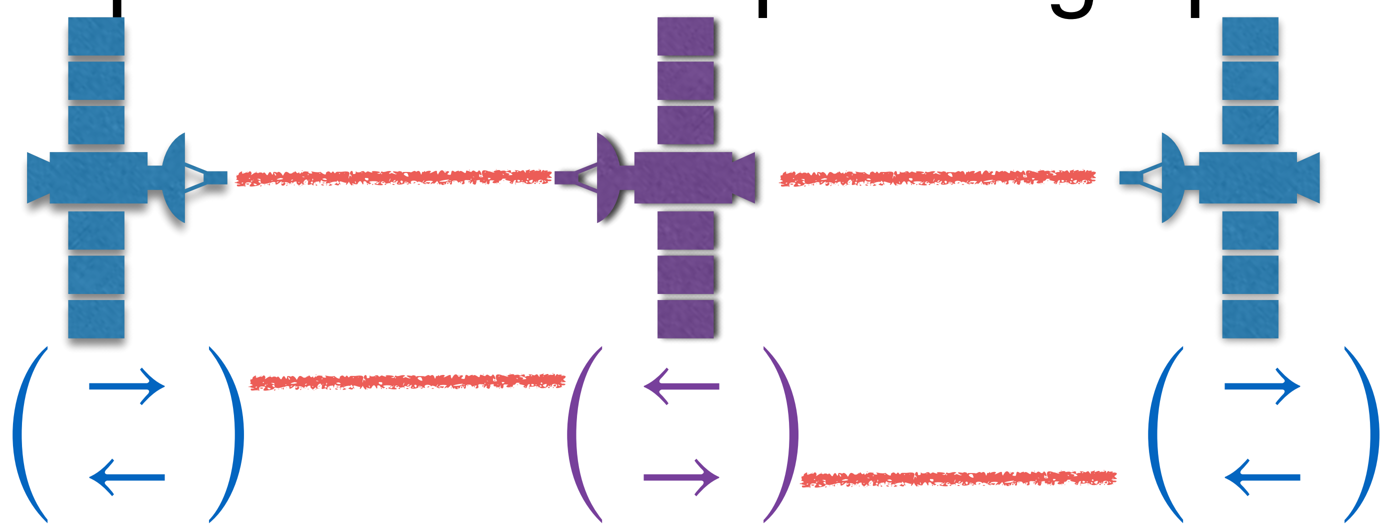
Bipartite Graphs



Every scan step is a bipartite graph!

---

2 steps suffice for bipartite graphs!



▶ **Observation 3.** *For instances of* MSC *in 1D for which the underlying graph G is bipartite, there exists a polynomial-time algorithm for computing an optimal scan cover.*

d.krupke@tu-bs.de

Minimum Cut Cover Problem

d.krupke@tu-bs.de

Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs

Technische Universität Braunschweig
d.krupke@tu-bs.de

Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

d.krupke@tu-bs.de

Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation
for scan cover in 1D

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation
for scan cover in 1D

Minimum Cut Cover Problem

Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

$$\boxed{\text{Min partition yields 2-approximation for scan cover in 1D}}$$

Minimum Cut Cover Problem $\overrightarrow{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]

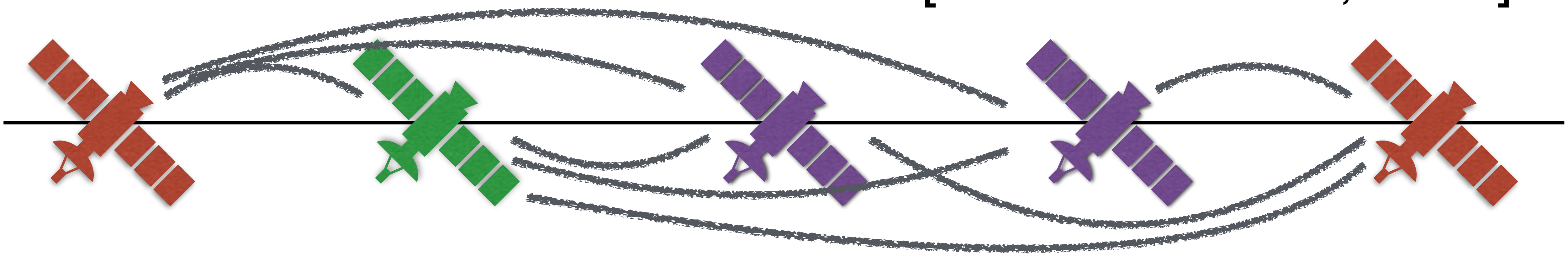
k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation
for scan cover in 1D

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]



Minimum Cut Cover Problem

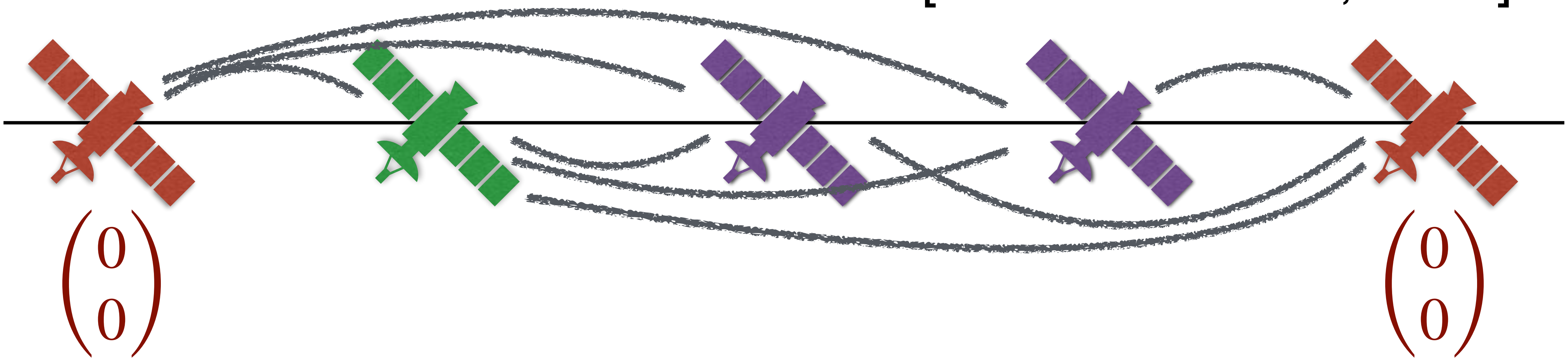
k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation for scan cover in 1D

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]



Minimum Cut Cover Problem

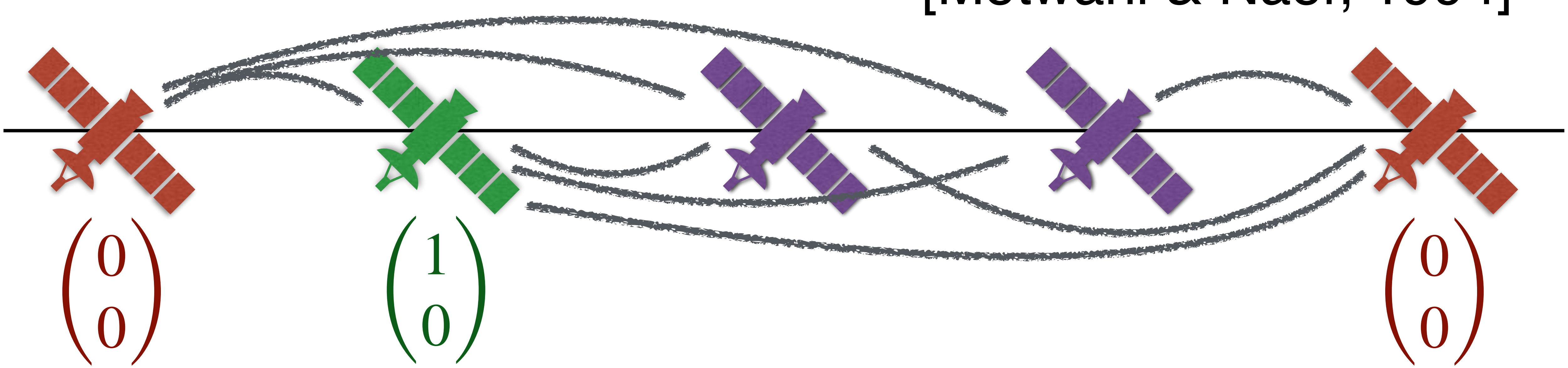
k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation for scan cover in 1D

Minimum Cut Cover Problem $\overrightarrow{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]



d.krupke@tu-bs.de

Minimum Cut Cover Problem

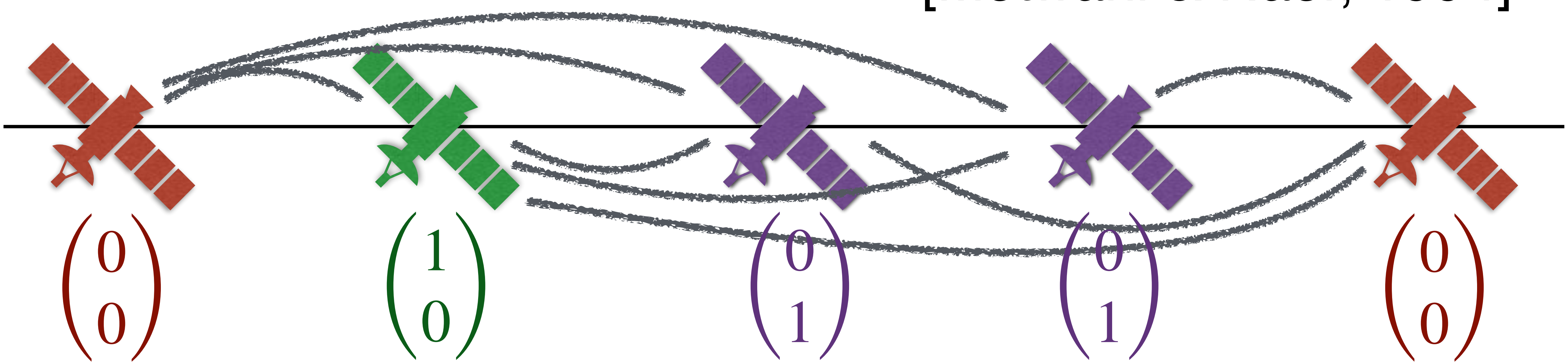
k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation for scan cover in 1D

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]



Minimum Cut Cover Problem

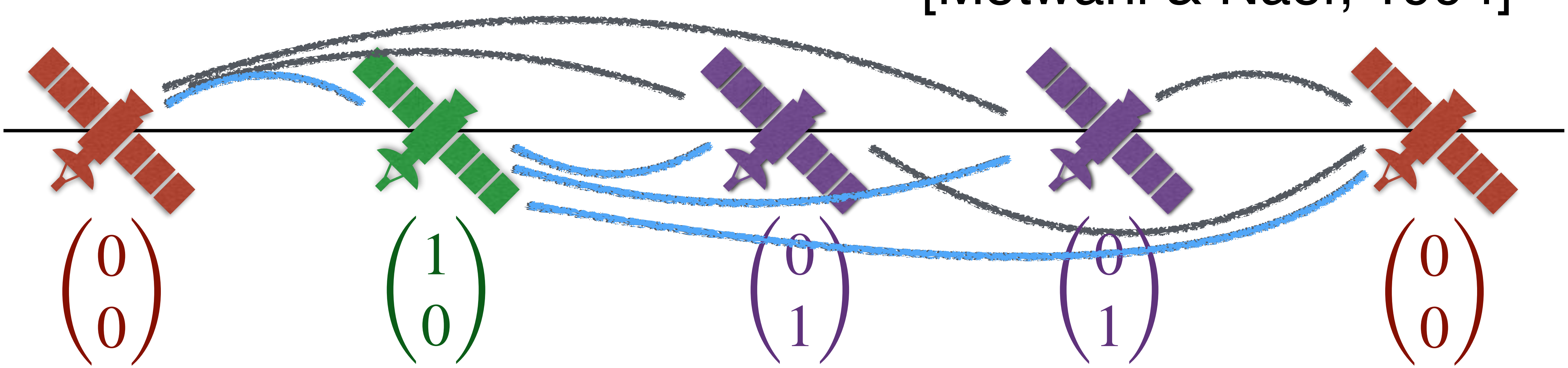
k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

$$\boxed{\text{Min partition yields 2-approximation for scan cover in 1D}}$$

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]



Minimum Cut Cover Problem

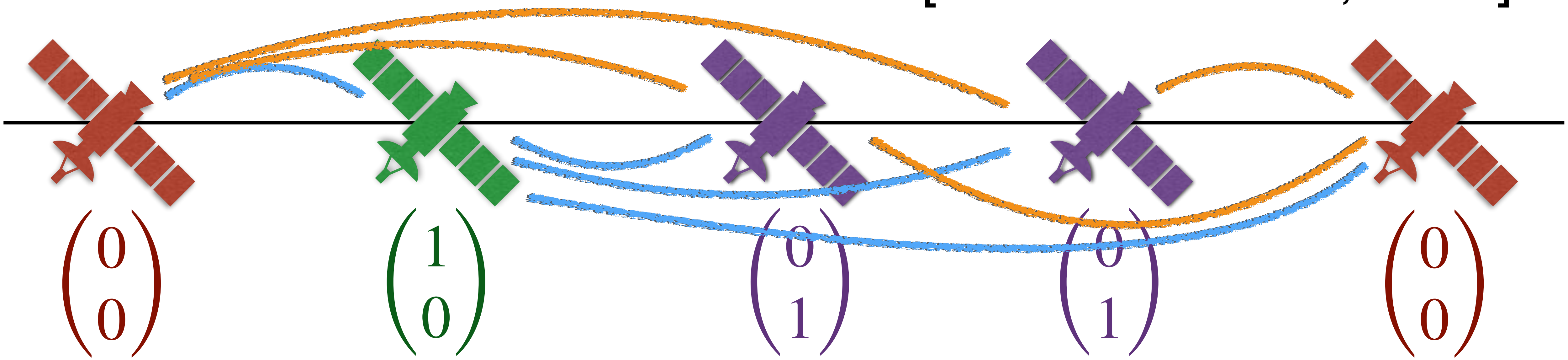
k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation for scan cover in 1D

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$

[Motwani & Naor, 1994]



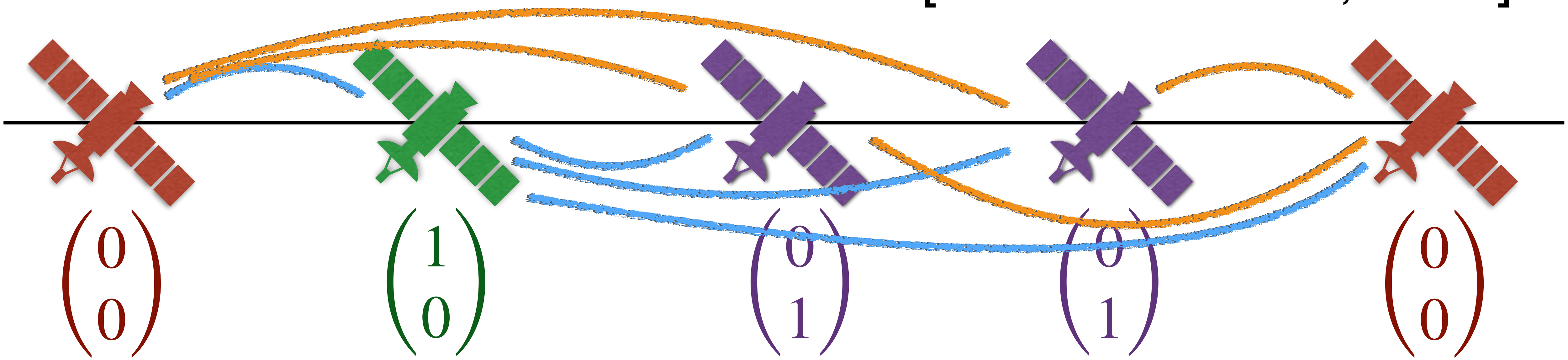
Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

Min partition yields 2-approximation for scan cover in 1D

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$ [Motwani & Naor, 1994]



1D-MSC: $\Theta(\lceil \log_2 \chi(G) \rceil)$

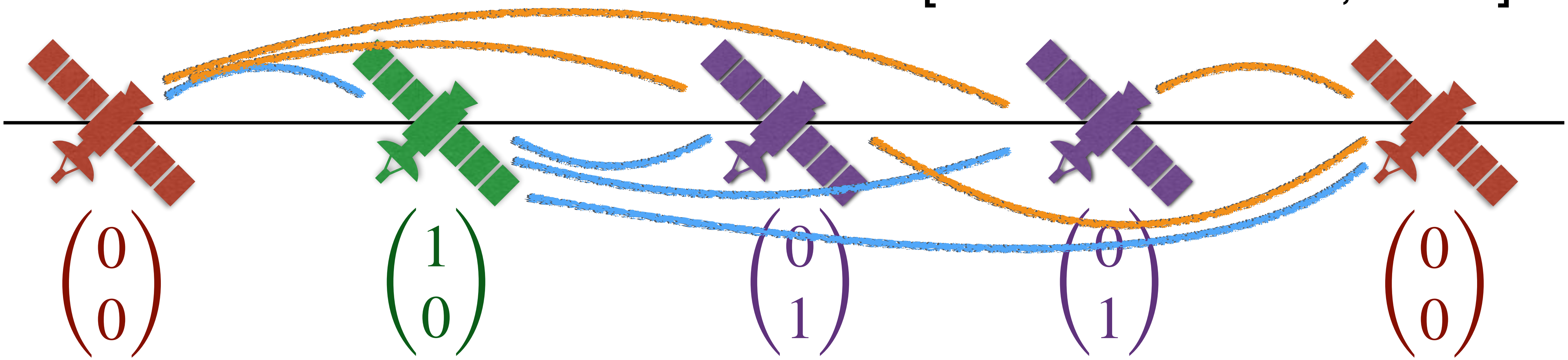
Minimum Cut Cover Problem

k scan steps $\Rightarrow$ Partition in $\leq k$ bip. graphs **LB!**

Partition in k bip. graphs $\Rightarrow$ $\leq 2k$ scan steps

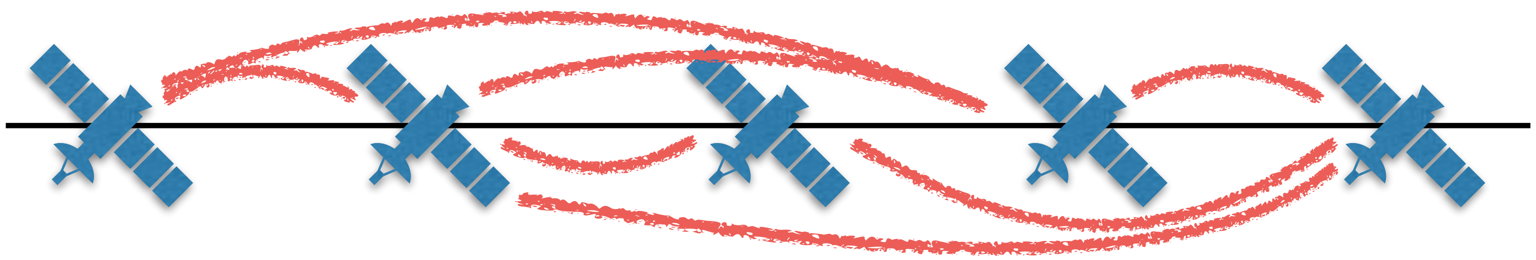
Min partition yields 2-approximation for scan cover in 1D

Minimum Cut Cover Problem $\vec{c}(G) = \lceil \log_2 \chi(G) \rceil$ [Motwani & Naor, 1994]



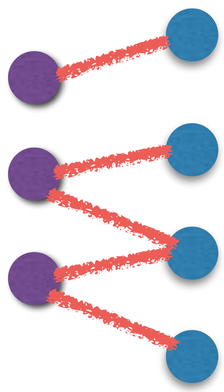
1D-MSC: $\Theta(\lceil \log_2 \chi(G) \rceil)$ log n makespans

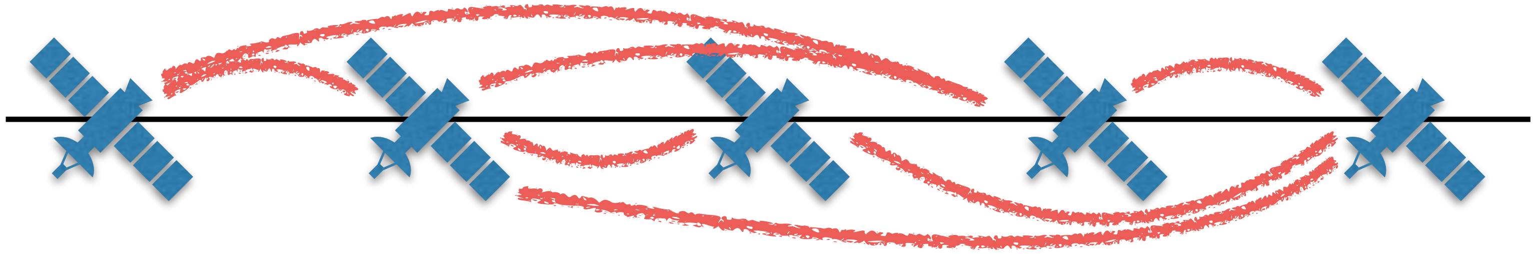
Directed Minimum Cut Cover



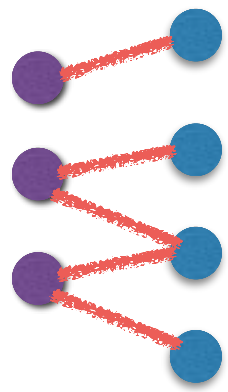
d.krupke@tu-bs.de

Directed Minimum Cut Cover

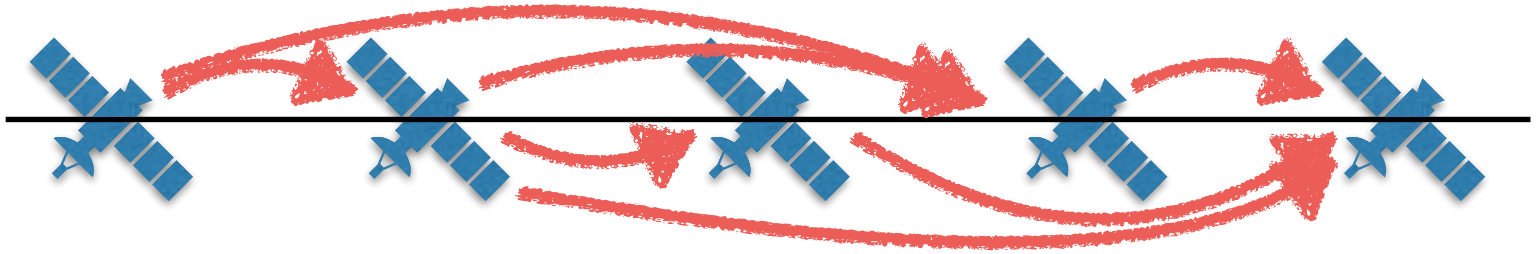




Directed Minimum Cut Cover

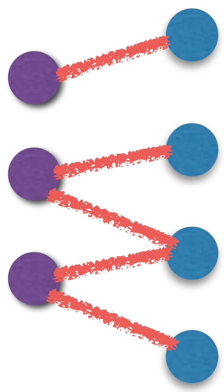


Cut

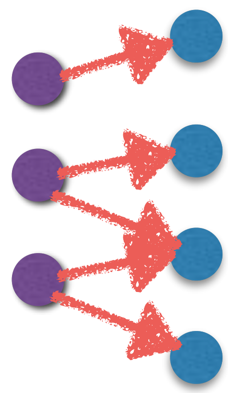


$\overrightarrow{G}$

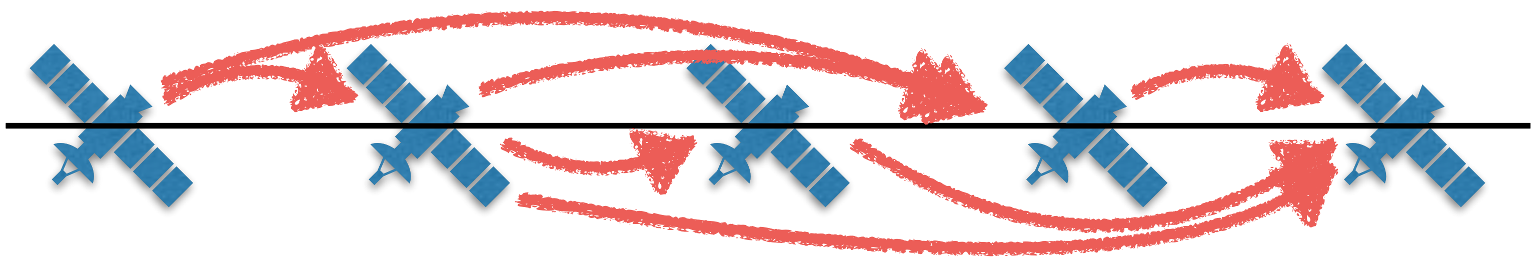
Directed Minimum Cut Cover



Cut

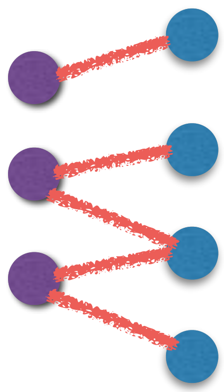


Directed Cut

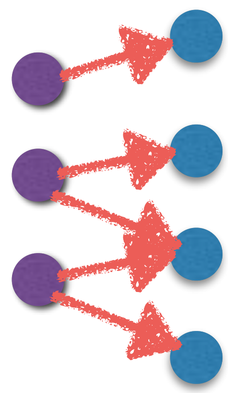


$\overrightarrow{G}$

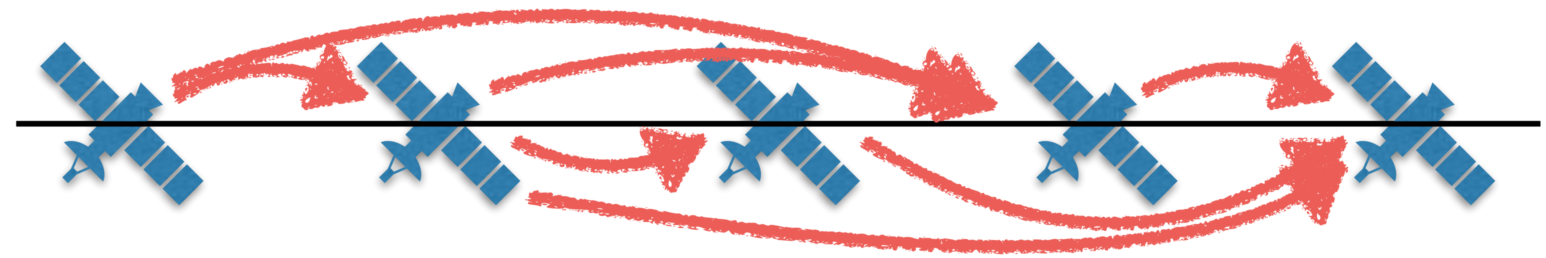
Directed Minimum Cut Cover



Cut



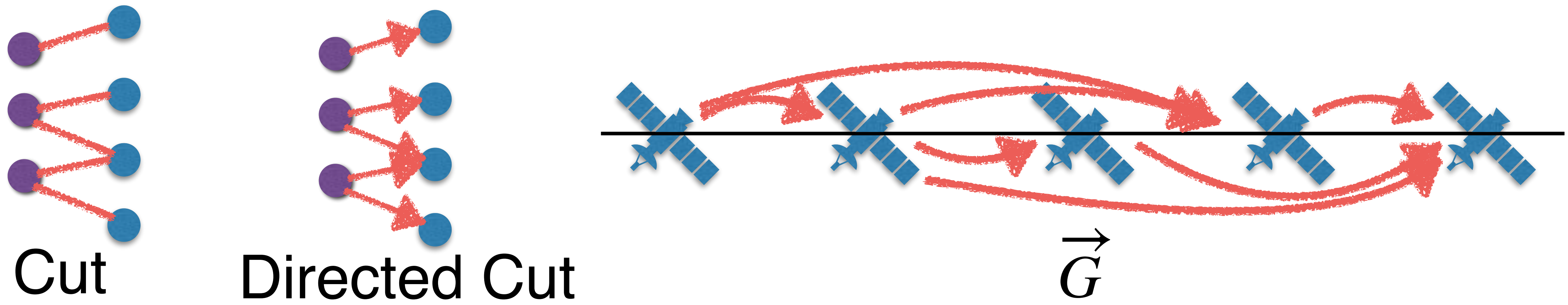
Directed Cut



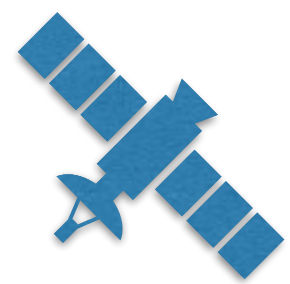
$\overrightarrow{G}$

Incoming XOR outgoing edges ⇒ single scan step

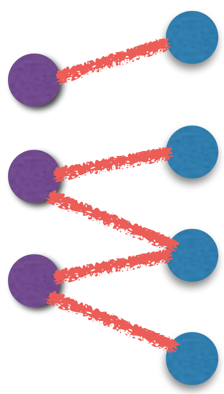
Directed Minimum Cut Cover



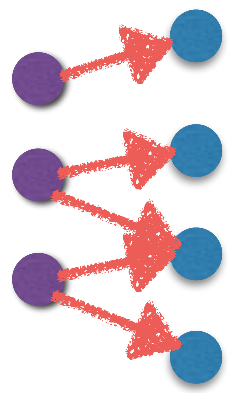
Incoming XOR outgoing edges ⇒ single scan step



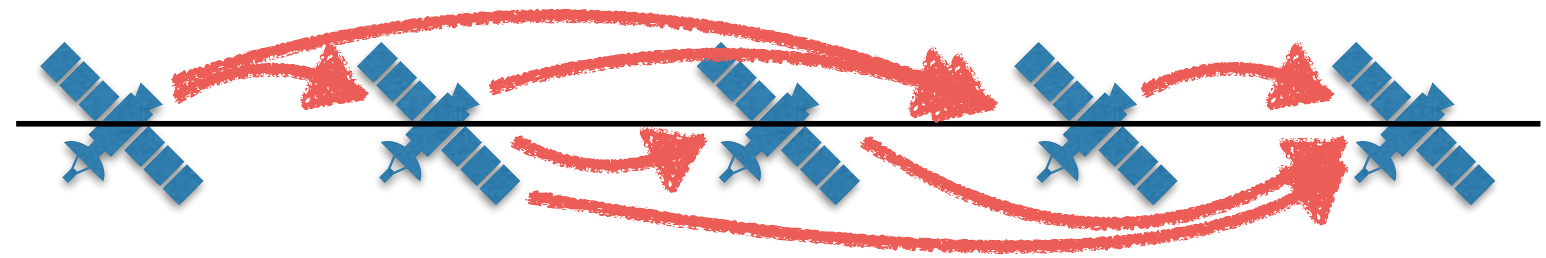
Directed Minimum Cut Cover



Cut

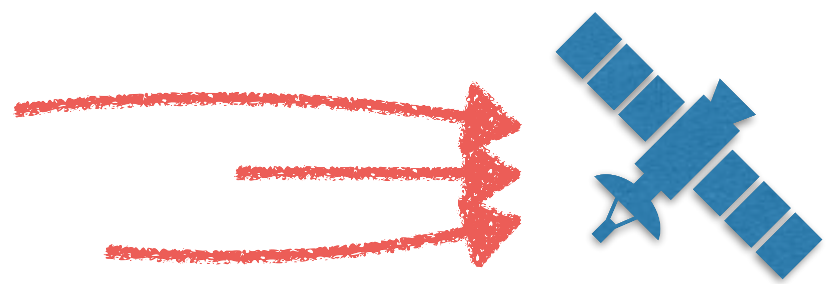


Directed Cut

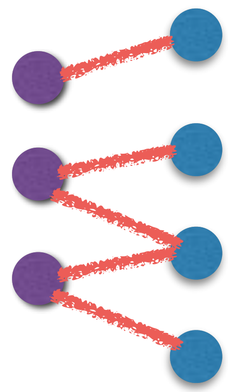


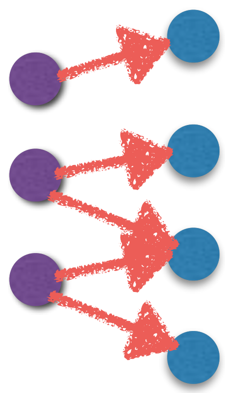
$\overrightarrow{G}$

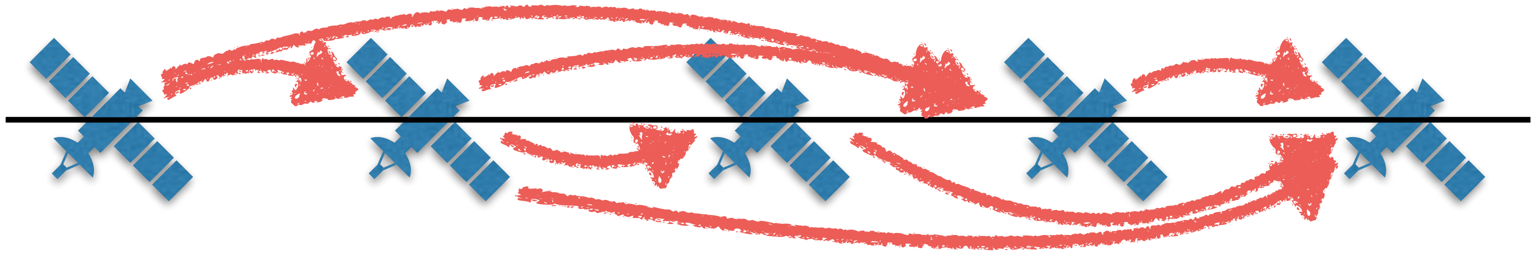
Incoming XOR outgoing edges ⇒ single scan step



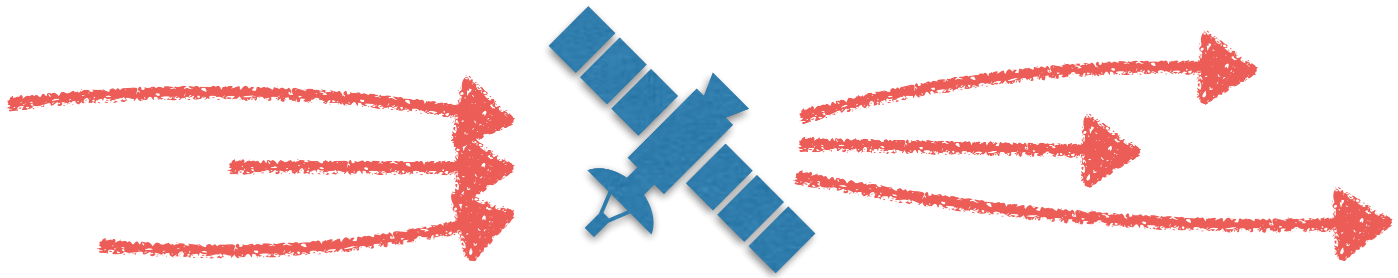
Directed Minimum Cut Cover


Cut

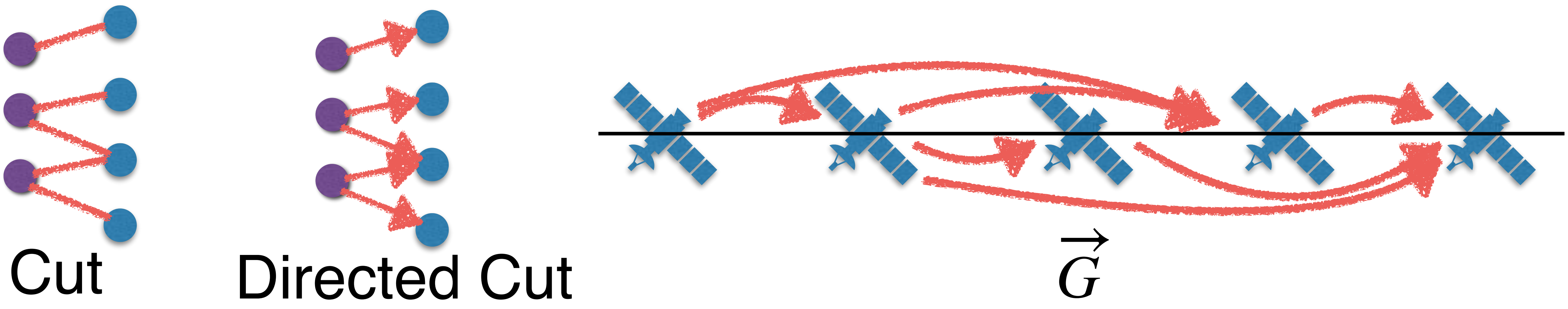

Directed Cut


$\overrightarrow{G}$

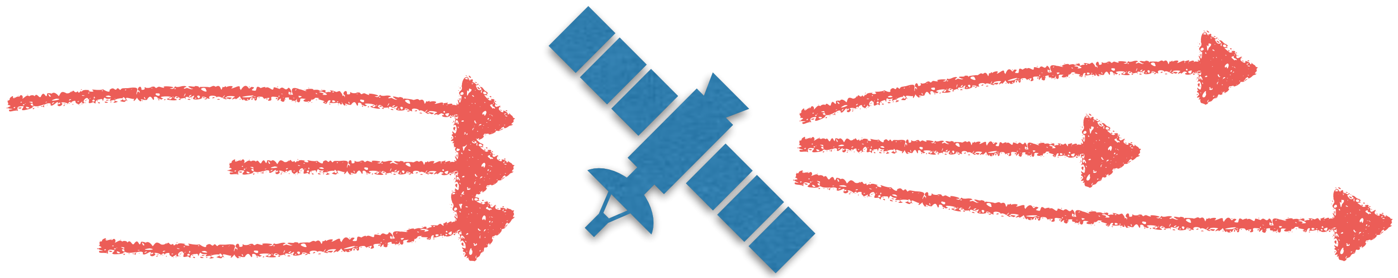
Incoming XOR outgoing edges ⇒ single scan step



Directed Minimum Cut Cover

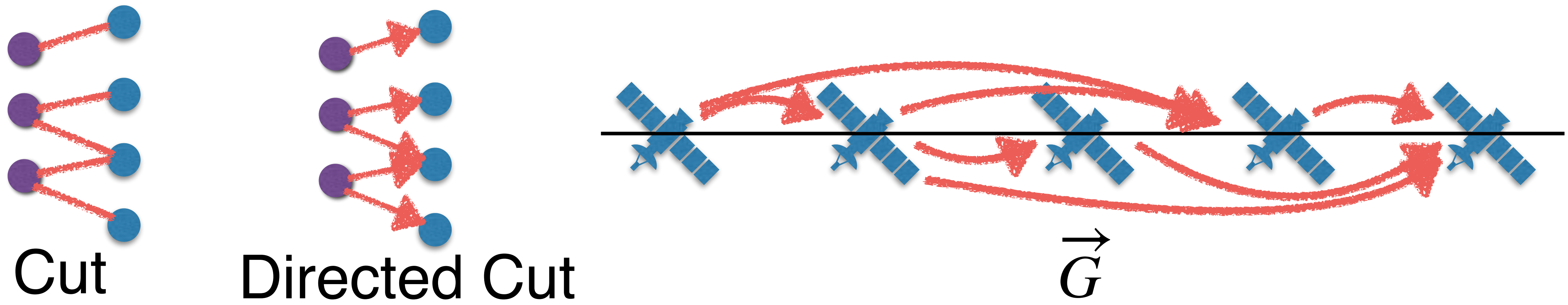


Incoming XOR outgoing edges ⇒ single scan step

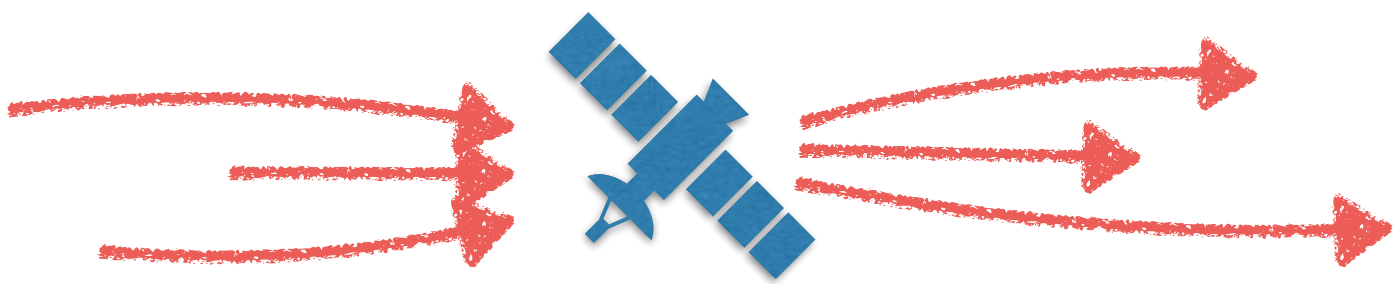


Partition in k directed cuts ⇔ k scan steps

Directed Minimum Cut Cover



Incoming XOR outgoing edges $\Rightarrow$ single scan step



Partition in k directed cuts $\Leftrightarrow$ k scan steps

$$\boxed{\overrightarrow{c}(\overrightarrow{G}) = \text{1D-MSC}(G)}$$

Directed Min Cut Cover Problem and $\chi(G)$

Directed Min Cut Cover Problem and $\chi(G)$

$$\lceil \log_2 \chi(G) \rceil = c(G)$$

Technische Universität Braunschweig
d.krupke@tu-bs.de

Directed Min Cut Cover Problem and $\chi(G)$

$$\lceil \log_2 \chi(G) \rceil = c(G) \leq \overrightarrow{c}(G)$$

Directed Min Cut Cover Problem and $\chi(G)$

$$\lceil \log_2 \chi(G) \rceil = c(G) \leq \overrightarrow{c}(G) \leq \lceil \log_2 \chi(G) \rceil + \lceil \log_2 \lceil \log_2 \chi(G) + 1 \rceil \rceil$$

[Watanabe et al., 1998]

Directed Min Cut Cover Problem and $\chi(G)$

$$\lceil \log_2 \chi(G) \rceil = c(G) \leq \vec{c}(G) \leq \lceil \log_2 \chi(G) \rceil + \lceil \log_2 \lceil \log_2 \chi(G) + 1 \rceil \rceil$$

[Watanabe et al., 1998]

▶ **Corollary 2.** *For every directed graph G, the directed cut cover number is bounded by*

$$\vec{c}(G) \leq \left\lceil \log_2 \chi(G) + \frac{1}{2} \log_2 \log_2 \chi(G) + 1 \right\rceil .$$

Directed Min Cut Cover Problem and $\chi(G)$

$$\lceil \log_2 \chi(G) \rceil = c(G) \leq \vec{c}(G) \leq \lceil \log_2 \chi(G) \rceil + \lceil \log_2 \lceil \log_2 \chi(G) + 1 \rceil \rceil$$

[Watanabe et al., 1998]

▶ **Corollary 2.** *For every directed graph G, the directed cut cover number is bounded by*

$$\vec{c}(G) \leq \left\lceil \log_2 \chi(G) + \frac{1}{2} \log_2 \log_2 \chi(G) + 1 \right\rceil.$$

▶ **Lemma 1.** *For every C, there exists a graph G and an ordering $<_L$ such that $\chi(G) > C$ and the number N of steps in every scan cover of $(G, <_L)$ is at least*

$$N \geq \left\lceil \log_2 \chi(G) + \frac{1}{4} \log_2 \log_2 \chi(G) \right\rceil.$$

No constant factor approximation!

No constant factor approximation!

▶ **Lemma 2.** *A C-approximation algorithm for* MSC *implies a polynomial-time algorithm for computing a coloring of graph G, $k := \chi(G)$, with $4^C \cdot k^C \cdot \sqrt{\log_2(k)}^C$ colors.*

No constant factor approximation!

▶ **Lemma 2.** *A C-approximation algorithm for* MSC *implies a polynomial-time algorithm for computing a coloring of graph G, $k := \chi(G)$, with $4^C \cdot k^C \cdot \sqrt{\log_2(k)}^C$ colors.*

▶ **Theorem 3.** *Even in 1D, a C-approximation for* MSC *for any $C \geq 1$ implies $P = NP$.*

No constant factor approximation!

▶ **Lemma 2.** *A C-approximation algorithm for* MSC *implies a polynomial-time algorithm for computing a coloring of graph G, $k := \chi(G)$, with $4^C \cdot k^C \cdot \sqrt{\log_2(k)}^C$ colors.*

▶ **Theorem 3.** *Even in 1D, a C-approximation for* MSC *for any $C \geq 1$ implies $P = NP$.*

Improved Inapproximability Results for MaxClique, Chromatic Number and Approximate Graph Coloring

Subhash Khot
Department of Computer Science
Princeton University
khot@cs.princeton.edu *

Abstract

In this paper, we present improved inapproximability results for three problems : the problem of finding the maximum clique size in a graph, the problem of finding the chromatic number of a graph, and the problem of coloring a graph with a small chromatic number with a small number of colors.

Håstad's celebrated result [13] shows that the maximum

1. Introduction

In this paper, we obtain improved inapproximability results for three problems, viz. the problem of finding the size of the largest clique in a graph, finding the chromatic number of a graph, and approximate coloring of a graph, i.e. coloring a graph with a small number of colors when the graph is guaranteed to have a small constant chromatic number. The first two results are obtained via new PCP con-

No constant factor approximation!

▶ **Lemma 2.** *A C-approximation algorithm for* MSC *implies a polynomial-time algorithm for computing a coloring of graph G, $k := \chi(G)$, with $4^C \cdot k^C \cdot \sqrt{\log_2(k)}^C$ colors.*

▶ **Theorem 3.** *Even in 1D, a C-approximation for* MSC *for any $C \geq 1$ implies $P = NP$.*

PCP inner verifier.

We also present a new hardness result for approximate graph coloring. We show that for all sufficiently large constants k, it is NP-hard to color a k-colorable graph with $k^{\frac{1}{25}(\log k)}$ colors. This improves a result of Fürer [11] that for arbitrarily small constant $\epsilon > 0$, for sufficiently large constants k, it is hard to color a k-colorable graph with $k^{3/2-\epsilon}$ colors.

Improved Inapproximability Results for MaxClique, Chromatic Number and Approximate Graph Coloring

Subhash Khot
Department of Computer Science
Princeton University
khot@cs.princeton.edu *

Abstract

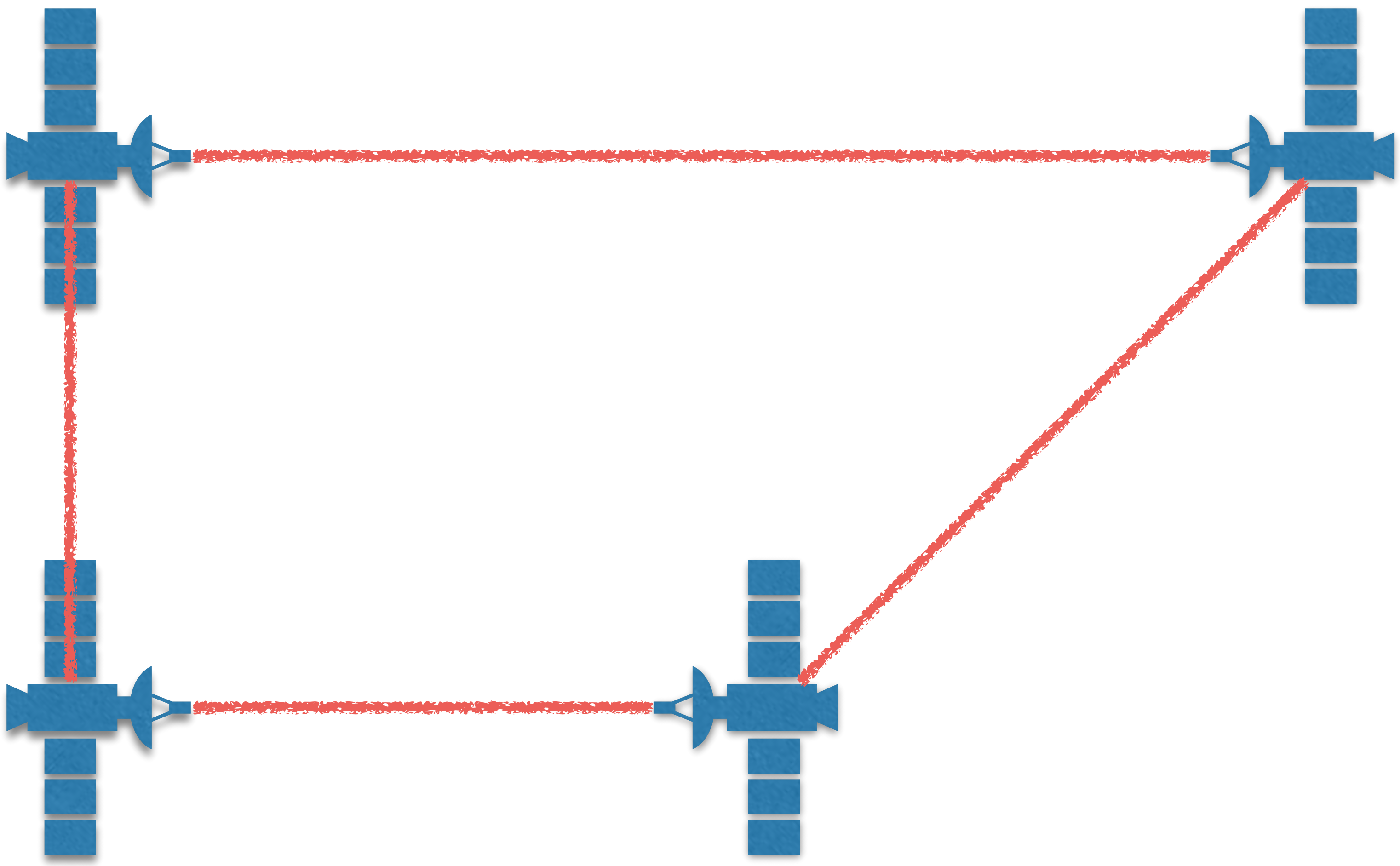
...aper, we present improved inapproximability re... ree problems : the problem of finding the maxi... size in a graph, the problem of finding the chro... ber of a graph, and the problem of coloring a ... a small chromatic number with a small number

...s celebrated result [13] shows that the maximum

1. Introduction

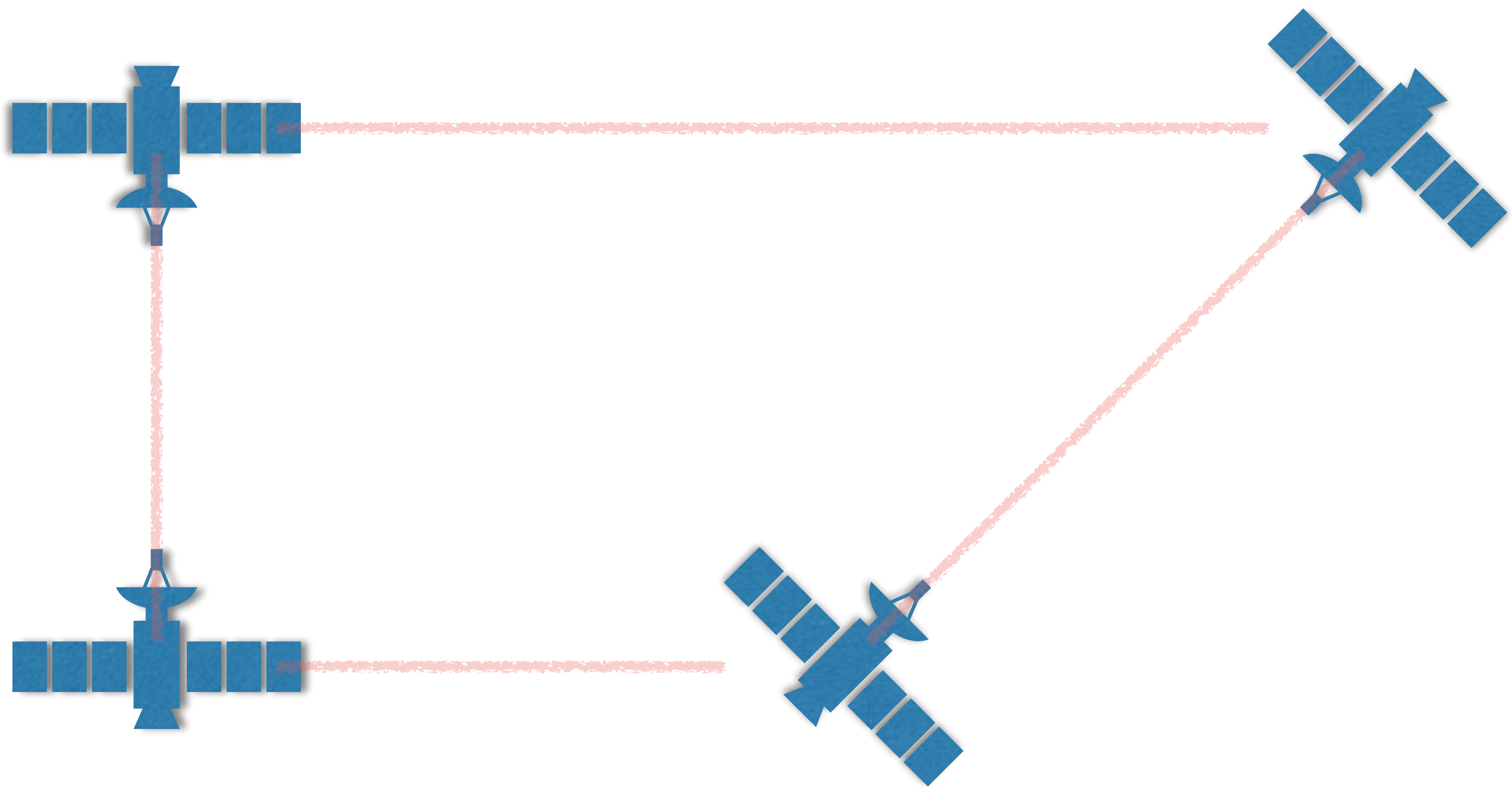
In this paper, we obtain improved inapproximability results for three problems, viz. the problem of finding the size of the largest clique in a graph, finding the chromatic number of a graph, and approximate coloring of a graph, i.e. coloring a graph with a small number of colors when the graph is guaranteed to have a small constant chromatic number. The first two results are obtained via new PCP con-

2-D



no longer discrete :(

2-D



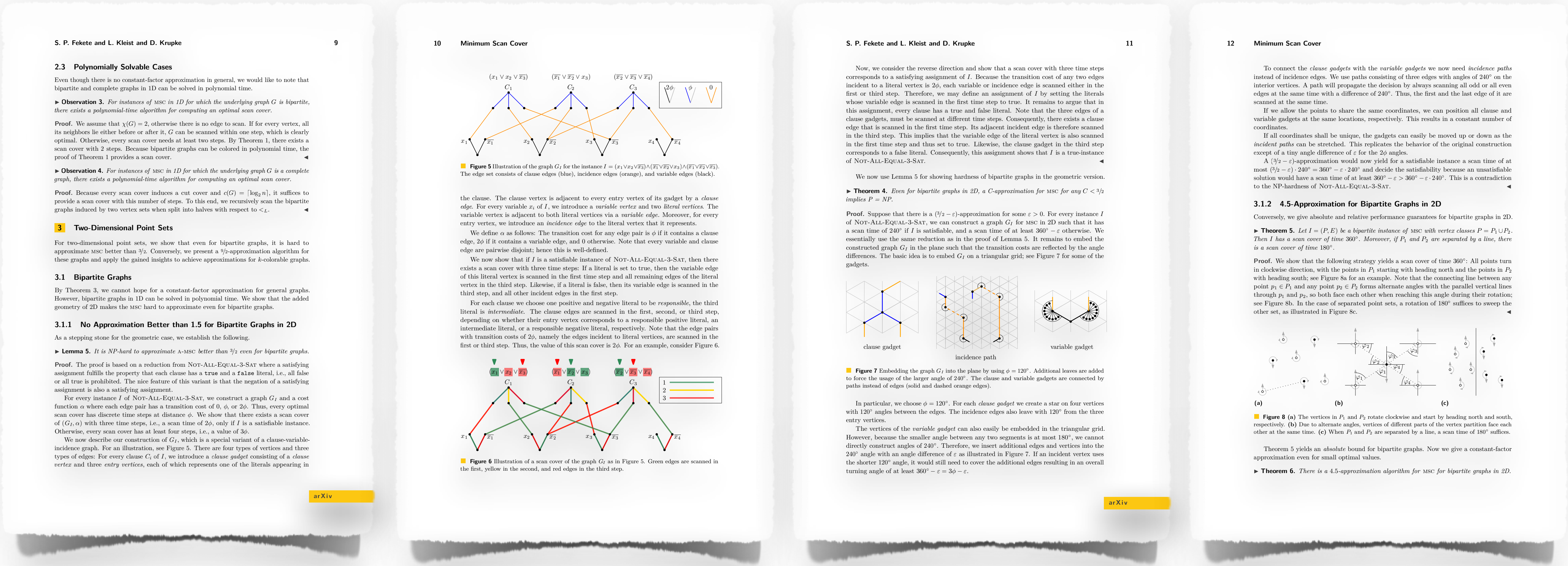
no longer discrete :(

Already bipartite graphs are hard

▶ **Theorem 4.** *Even for bipartite graphs in 2D, a C-approximation for* MSC *for any* $C < 3/2$ *implies* $P = NP$.

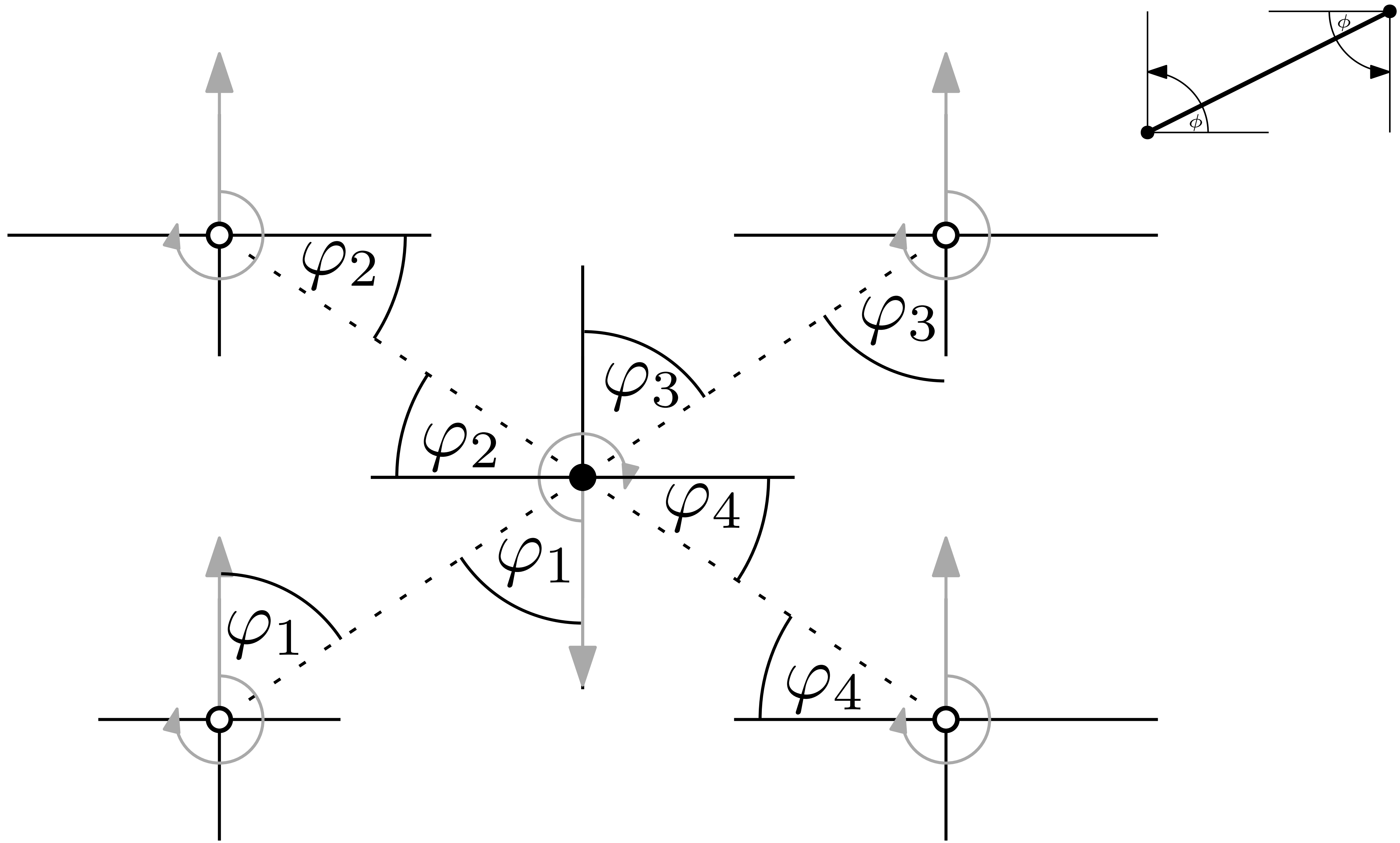
Already bipartite graphs are hard

▶ **Theorem 4.** *Even for bipartite graphs in 2D, a C-approximation for* MSC *for any $C < 3/2$ implies P = NP.*

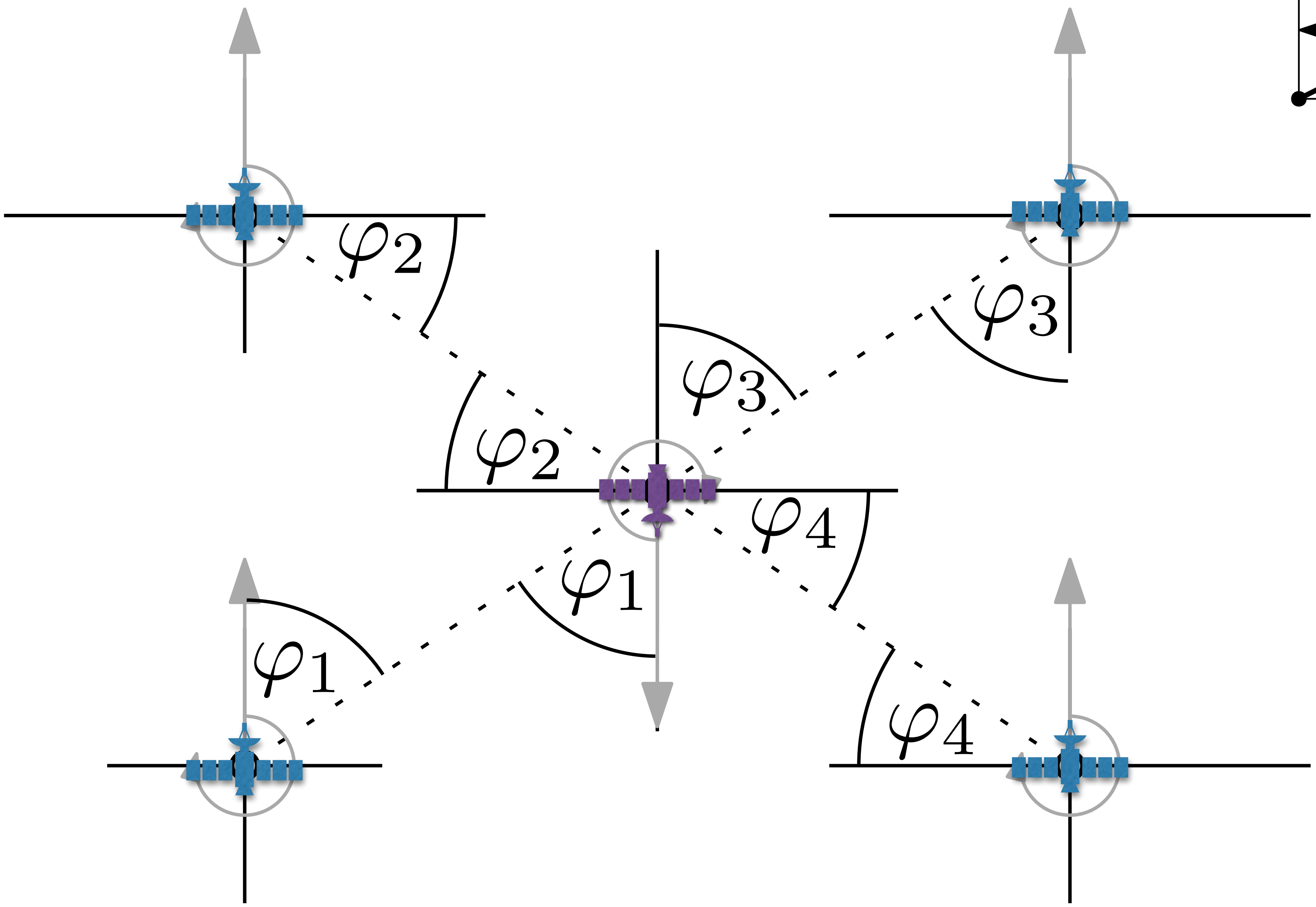


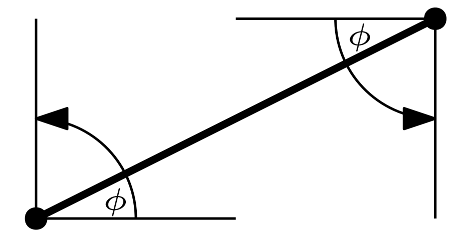
d.krupke@tu-bs.de

But: Alternating Angles



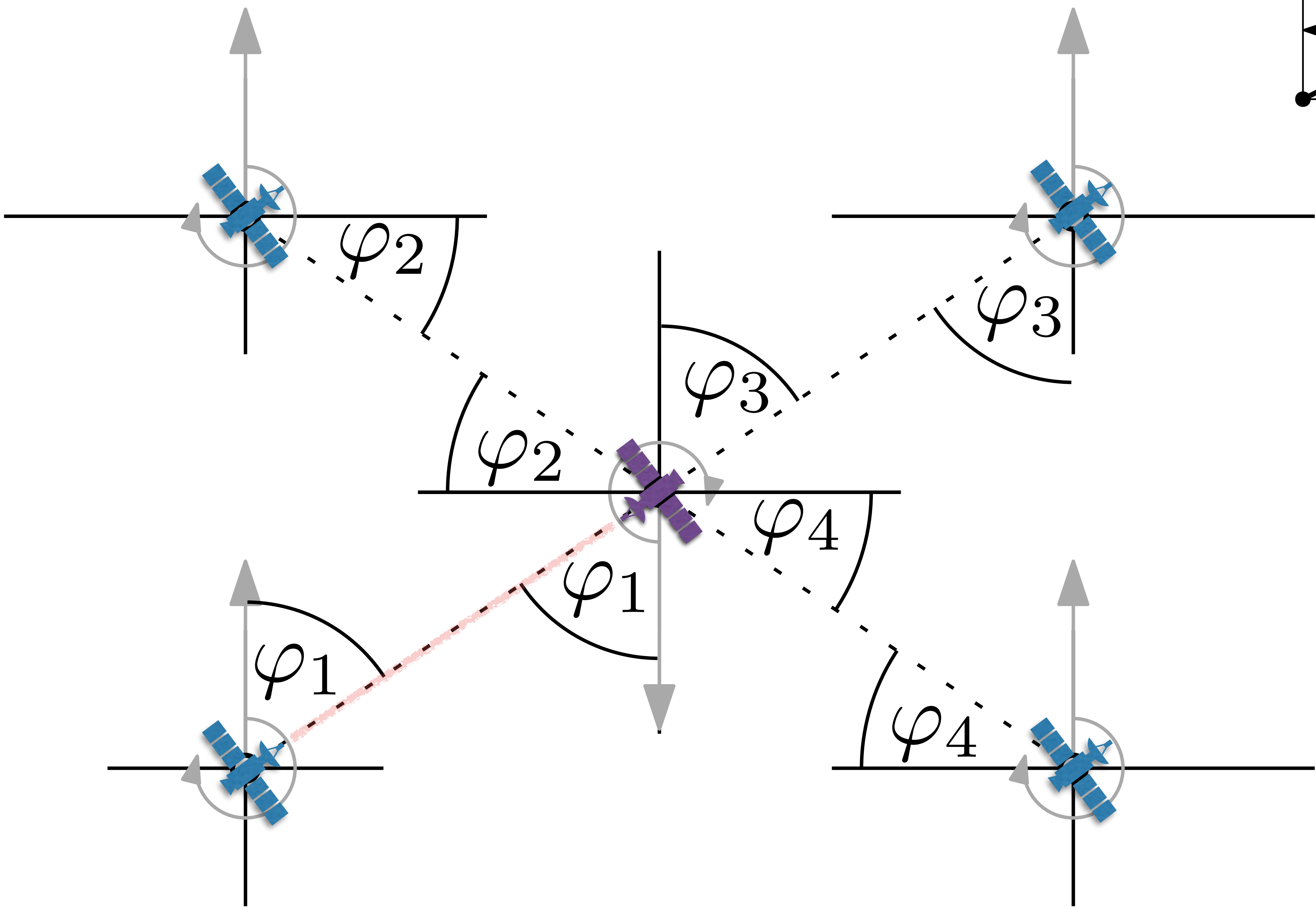
But: Alternating Angles

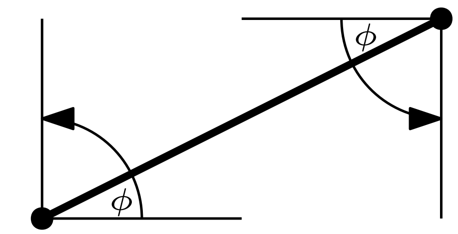




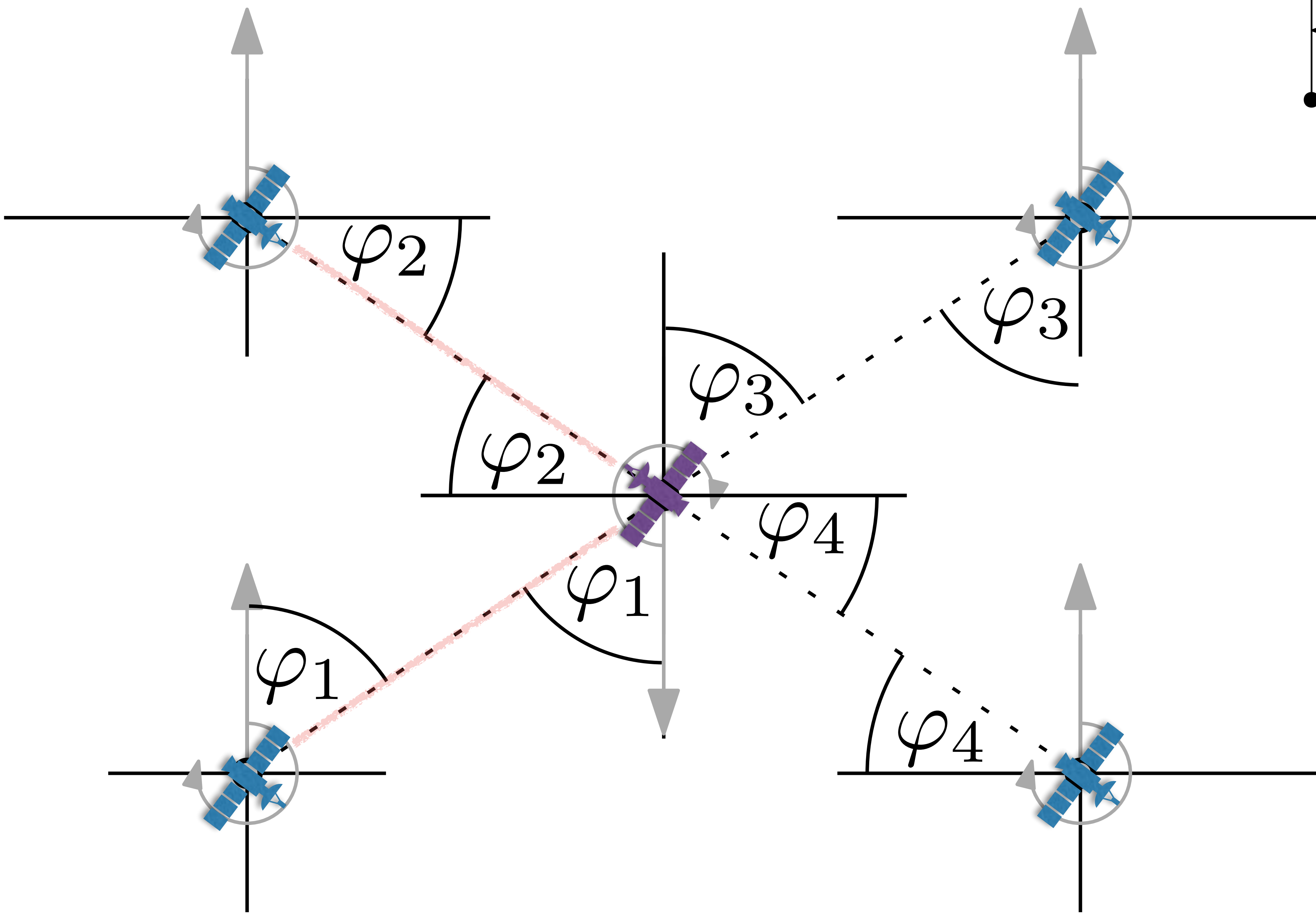
d.krupke@tu-bs.de

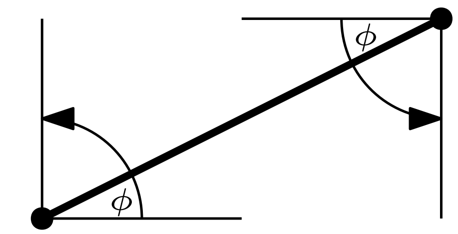
But: Alternating Angles



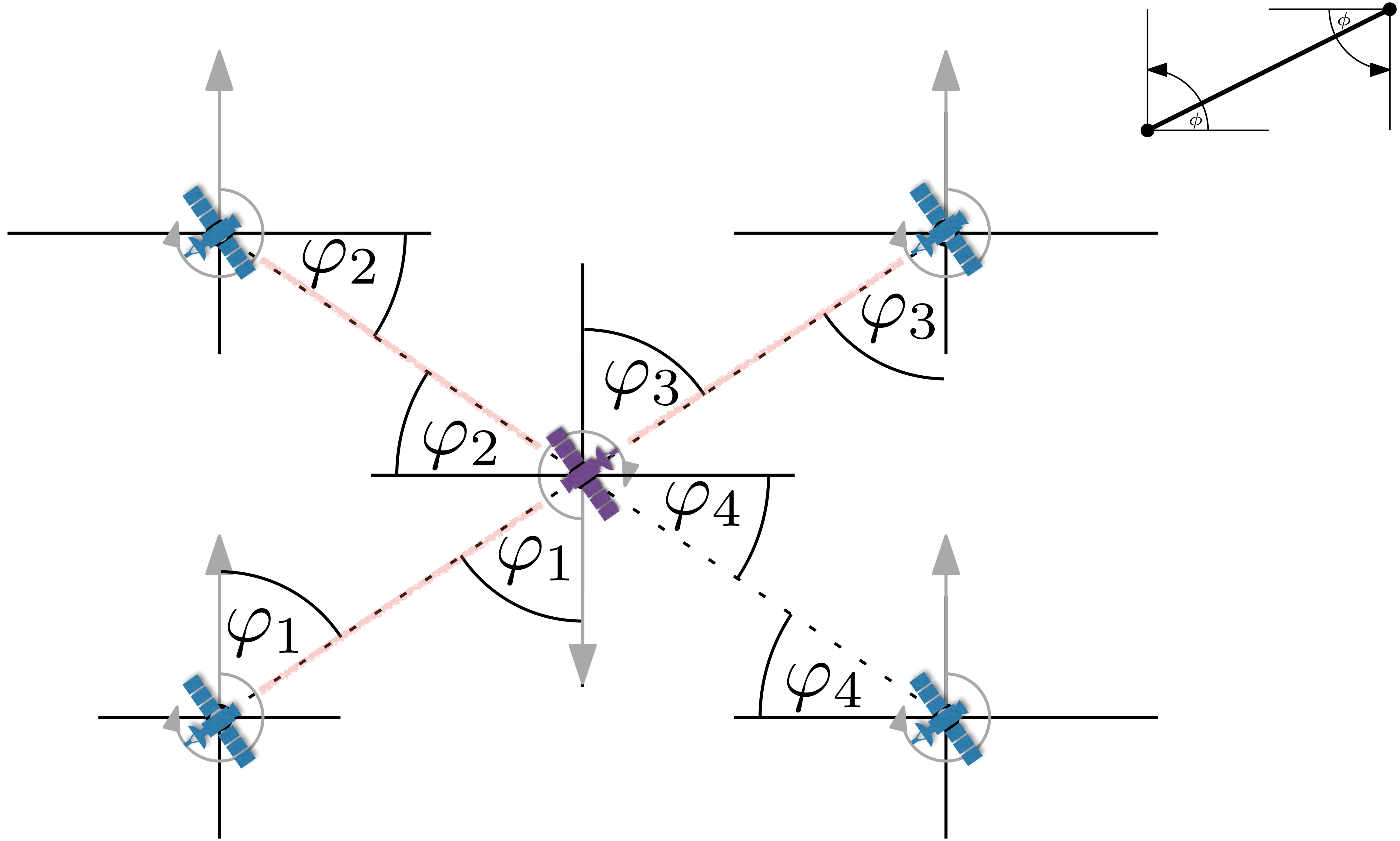


But: Alternating Angles

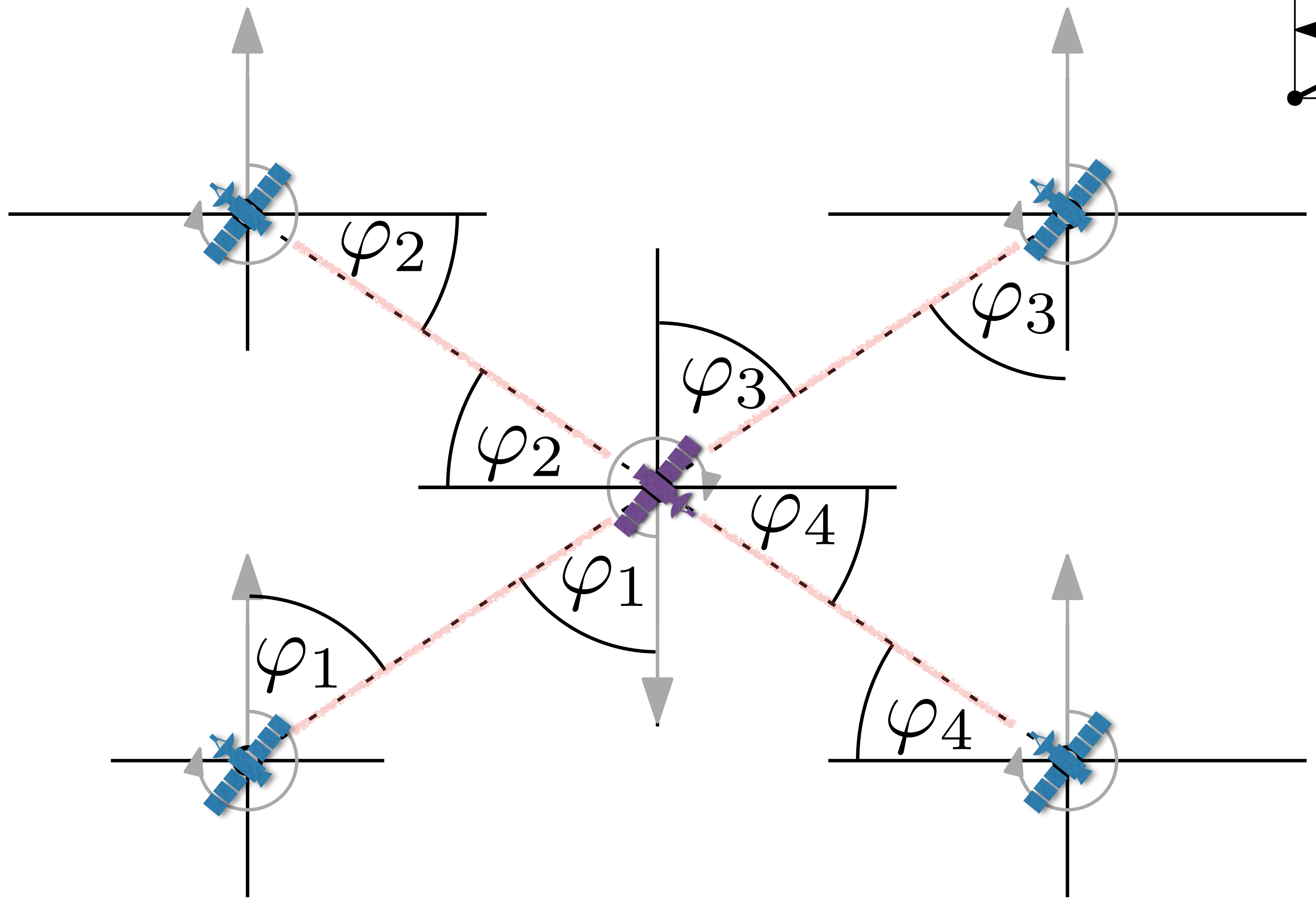


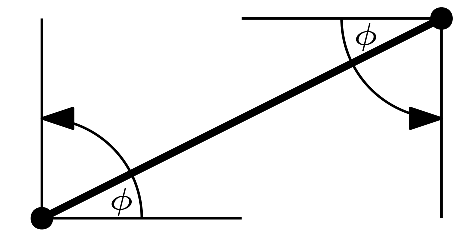


But: Alternating Angles



But: Alternating Angles





d.krupke@tu-bs.de

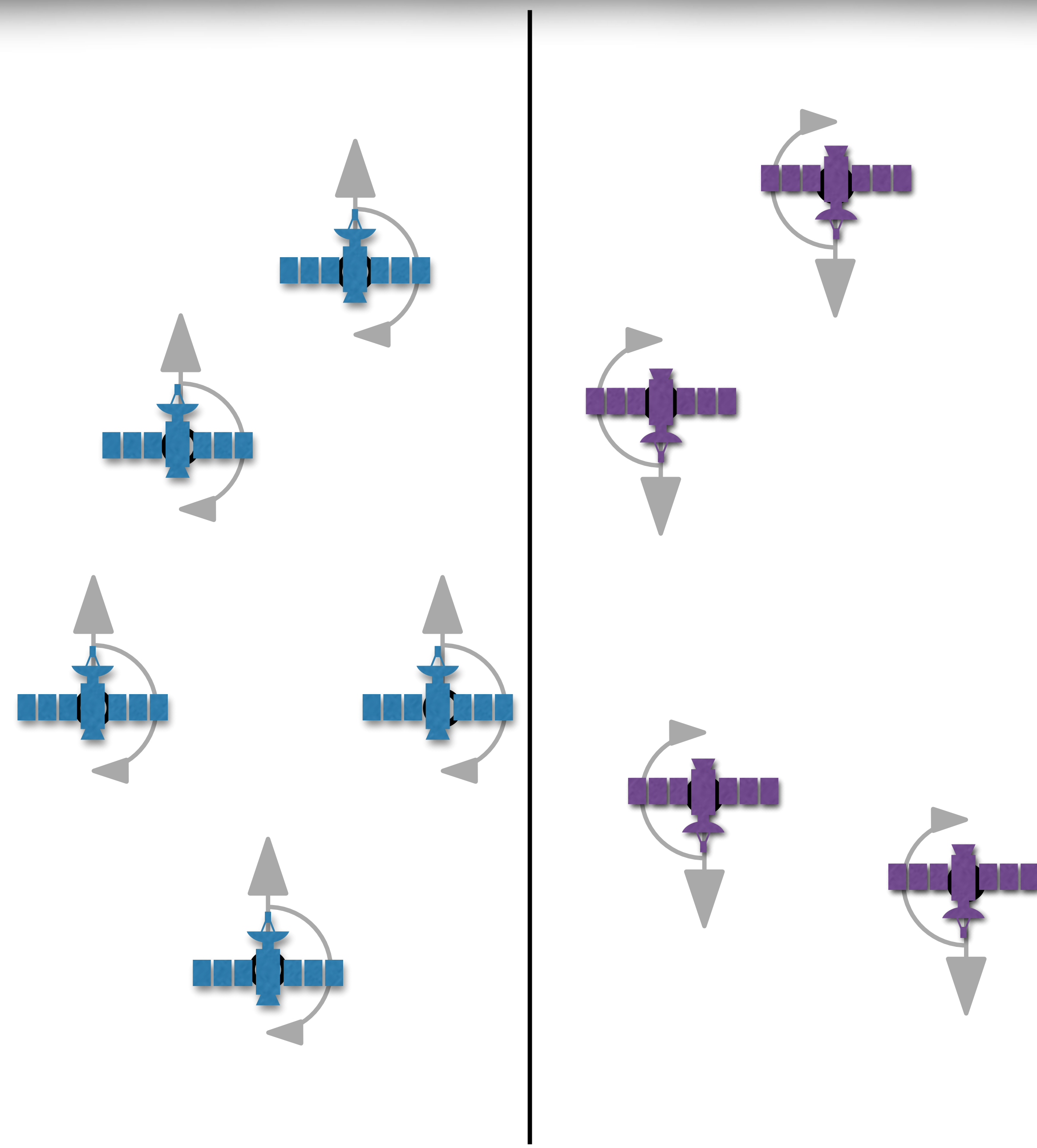
Scanning bipartite graphs in constant time

▶ **Theorem 5.** *Let $I = (P, E)$ be a bipartite instance of* MSC *with vertex classes $P = P_1 \cup P_2$. Then I has a scan cover of time $360°$. Moreover, if P_1 and P_2 are separated by a line, there is a scan cover of time $180°$.*

d.krupke@tu-bs.de

Scanning bipartite graphs in constant time

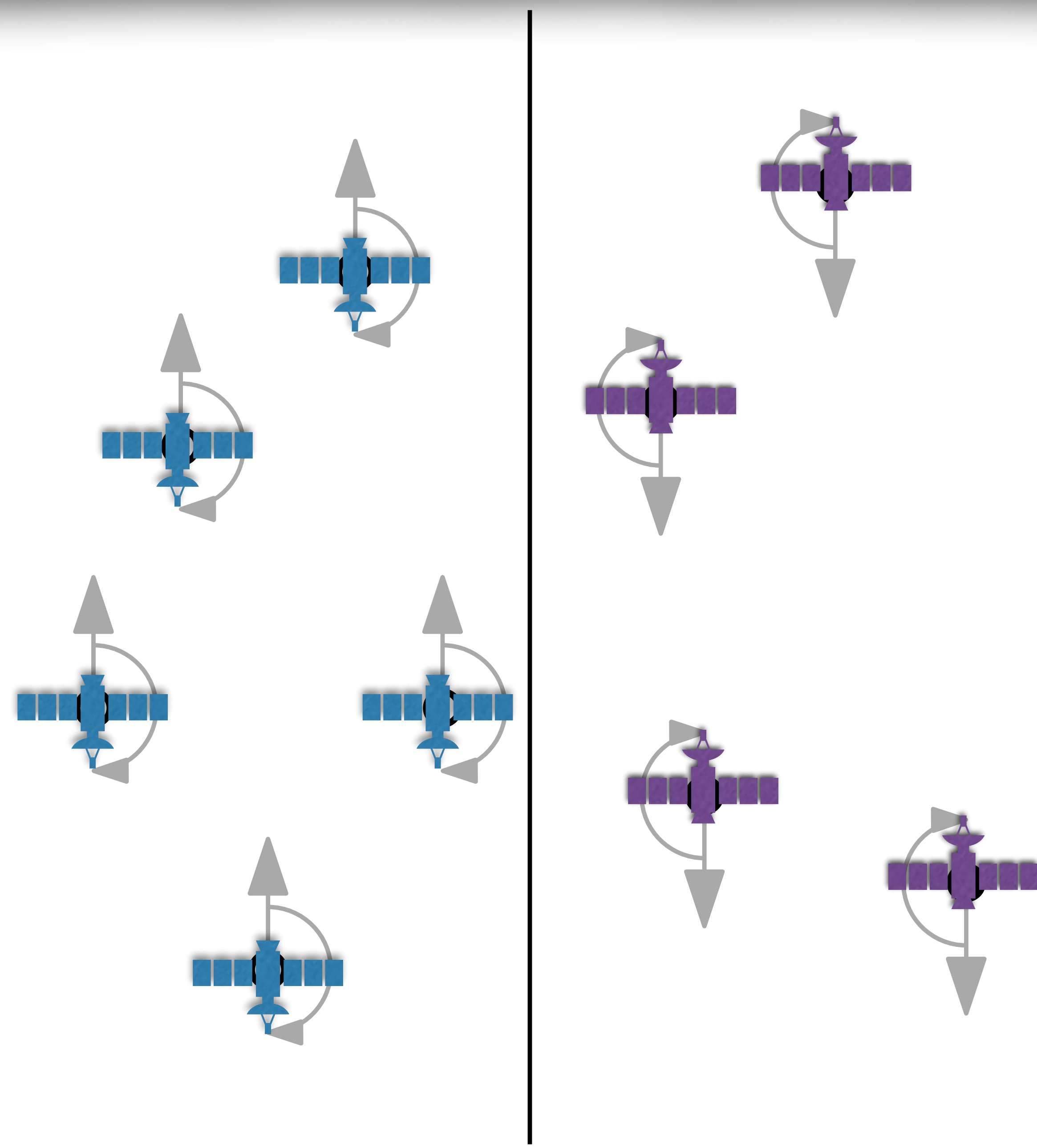
▶ **Theorem 5.** *Let $I = (P, E)$ be a bipartite instance of* MSC *with vertex classes $P = P_1 \cup P_2$. Then I has a scan cover of time $360°$. Moreover, if P_1 and P_2 are separated by a line, there is a scan cover of time $180°$.*



d.krupke@tu-bs.de

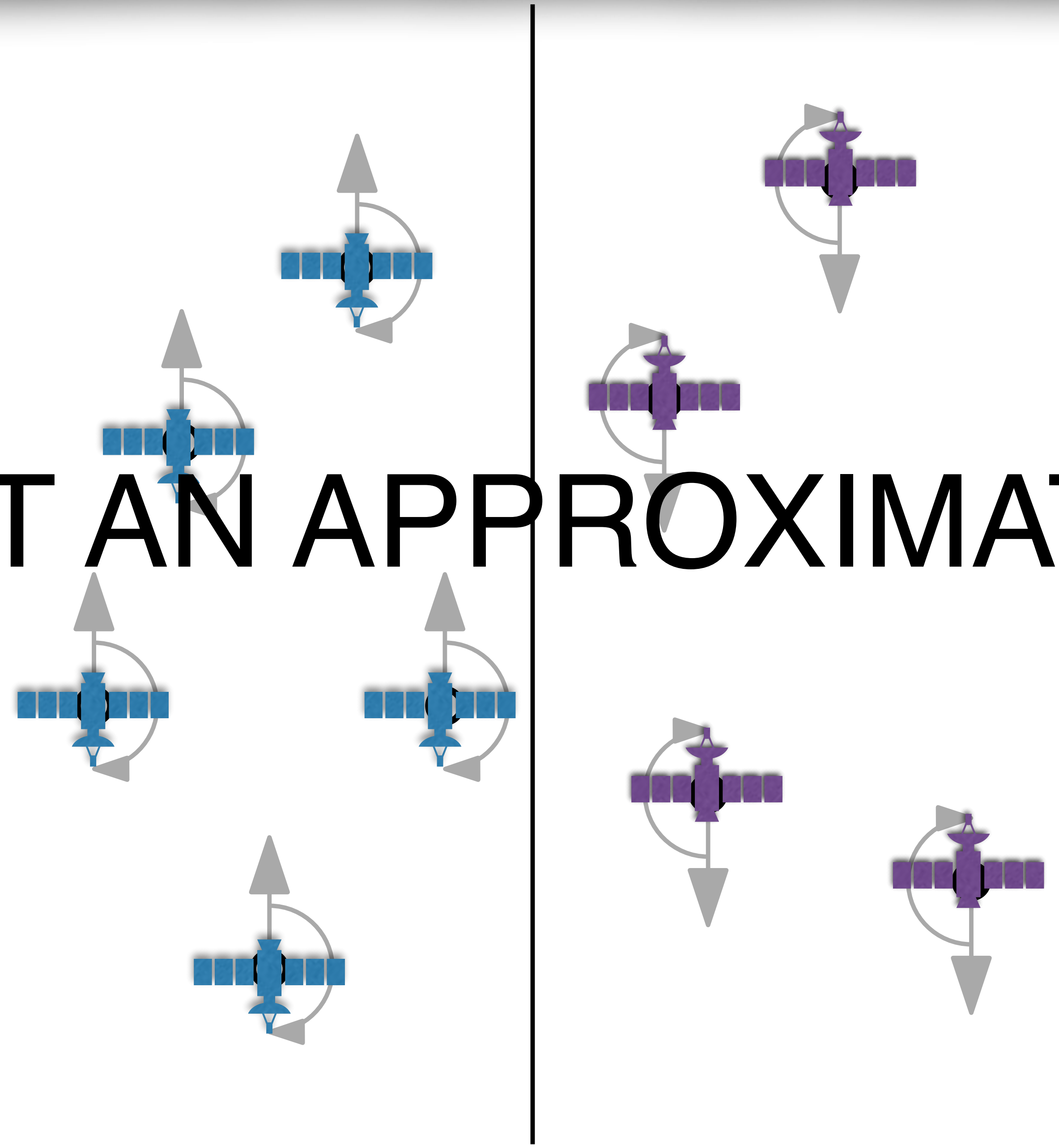
Scanning bipartite graphs in constant time

▶ **Theorem 5.** *Let $I = (P, E)$ be a bipartite instance of* MSC *with vertex classes $P = P_1 \cup P_2$. Then I has a scan cover of time $360°$. Moreover, if P_1 and P_2 are separated by a line, there is a scan cover of time $180°$.*



Scanning bipartite graphs in constant time

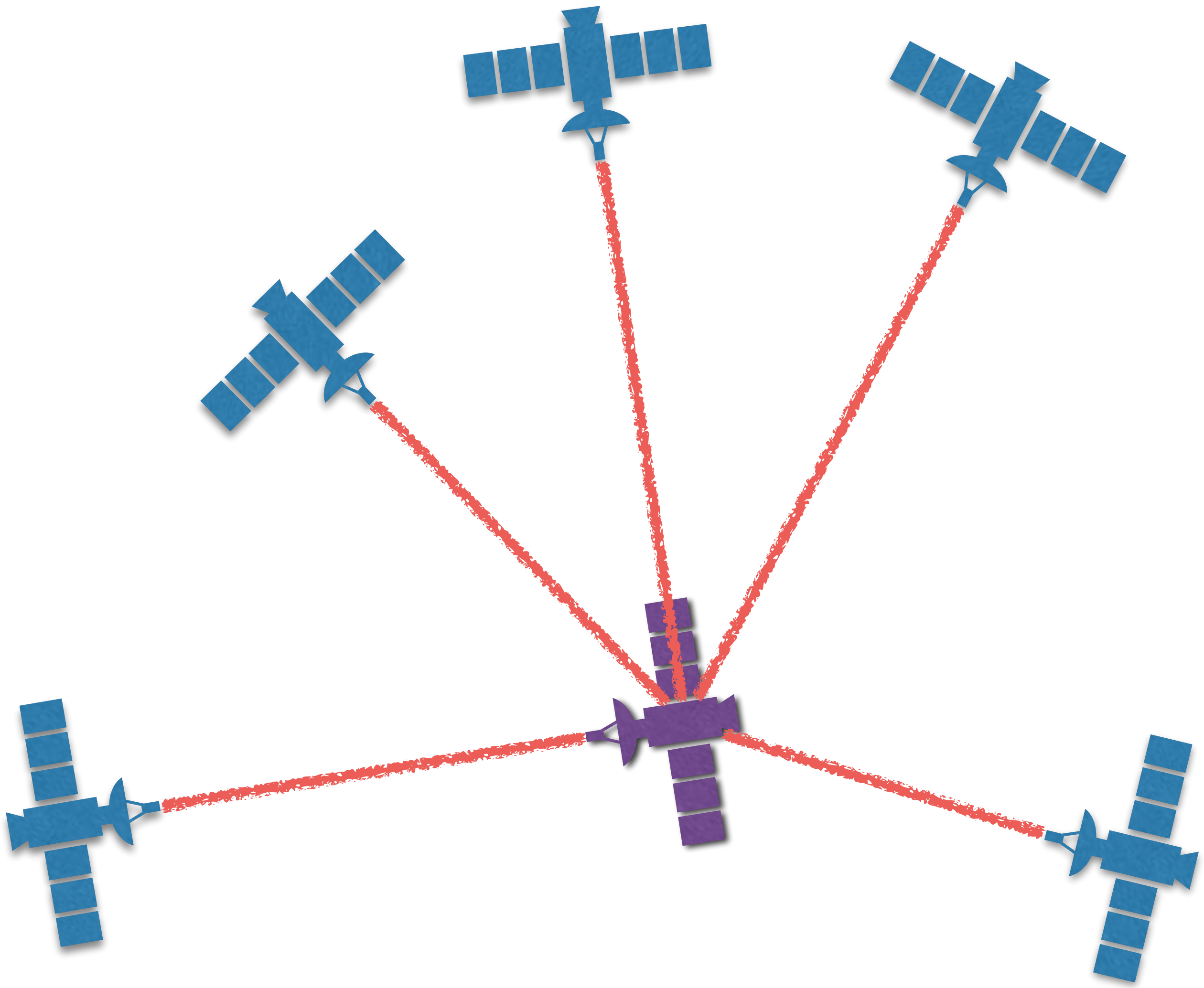
▶ **Theorem 5.** *Let $I = (P, E)$ be a bipartite instance of* MSC *with vertex classes $P = P_1 \cup P_2$. Then I has a scan cover of time $360°$. Moreover, if P_1 and P_2 are separated by a line, there is a scan cover of time $180°$.*



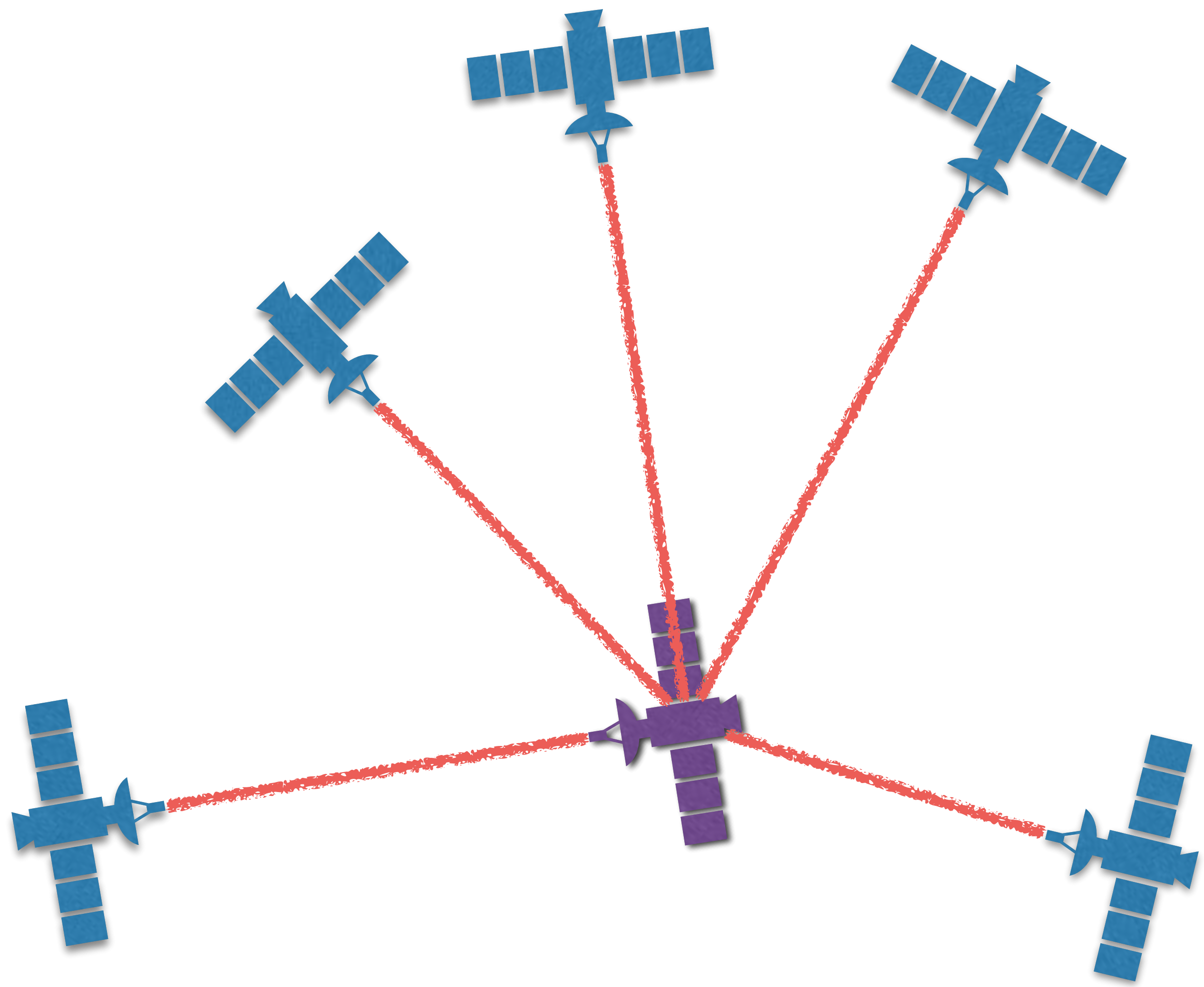
NOT AN APPROXIMATION!

d.krupke@tu-bs.de

A simple lower bound



A simple lower bound

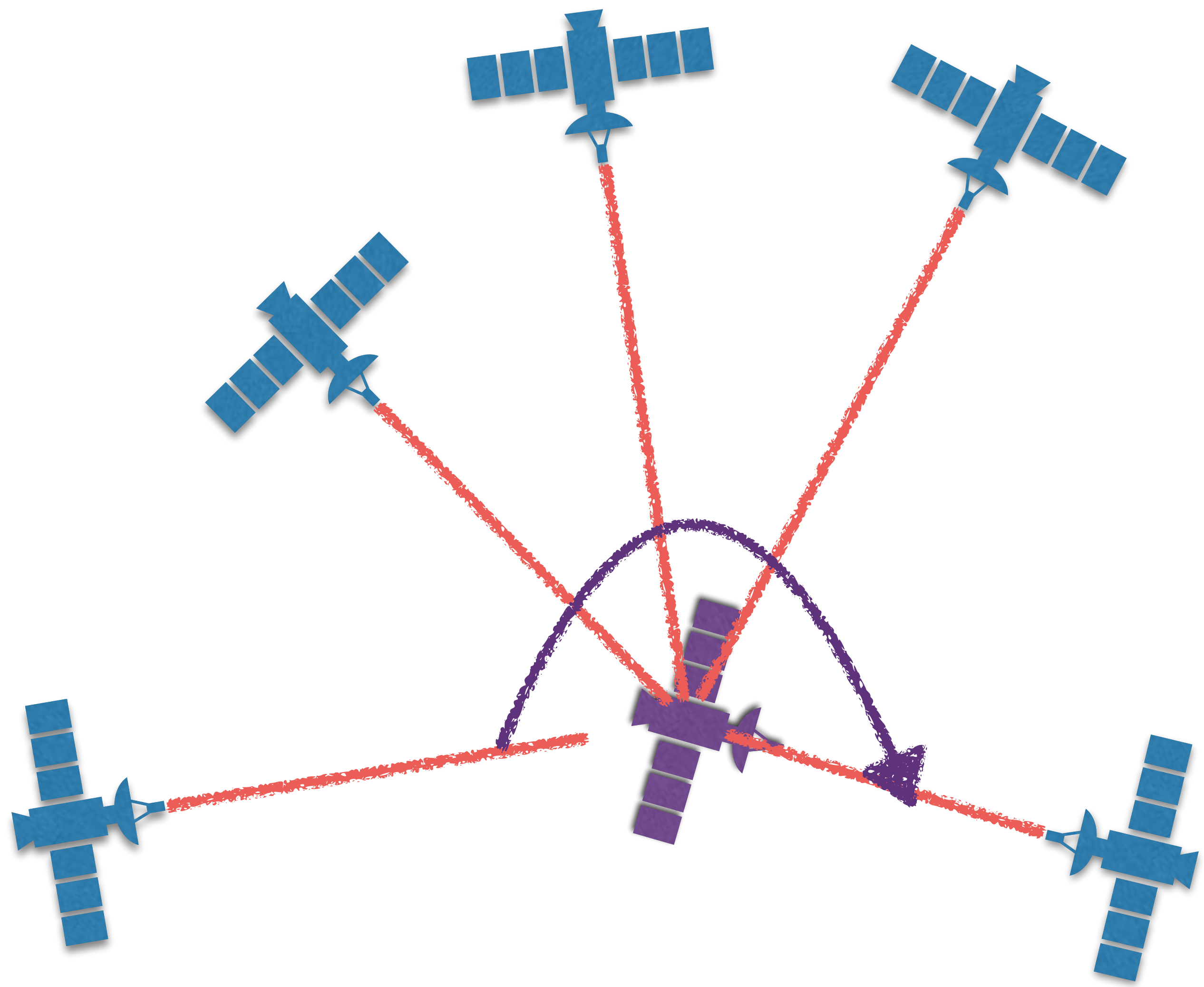


$$\max_{s \in V} \text{optimal solution of } s \text{ to cover neighbors}$$

Sándor Fekete, Linda Kleist, and Dominik Krupke

d.krupke@tu-bs.de

A simple lower bound

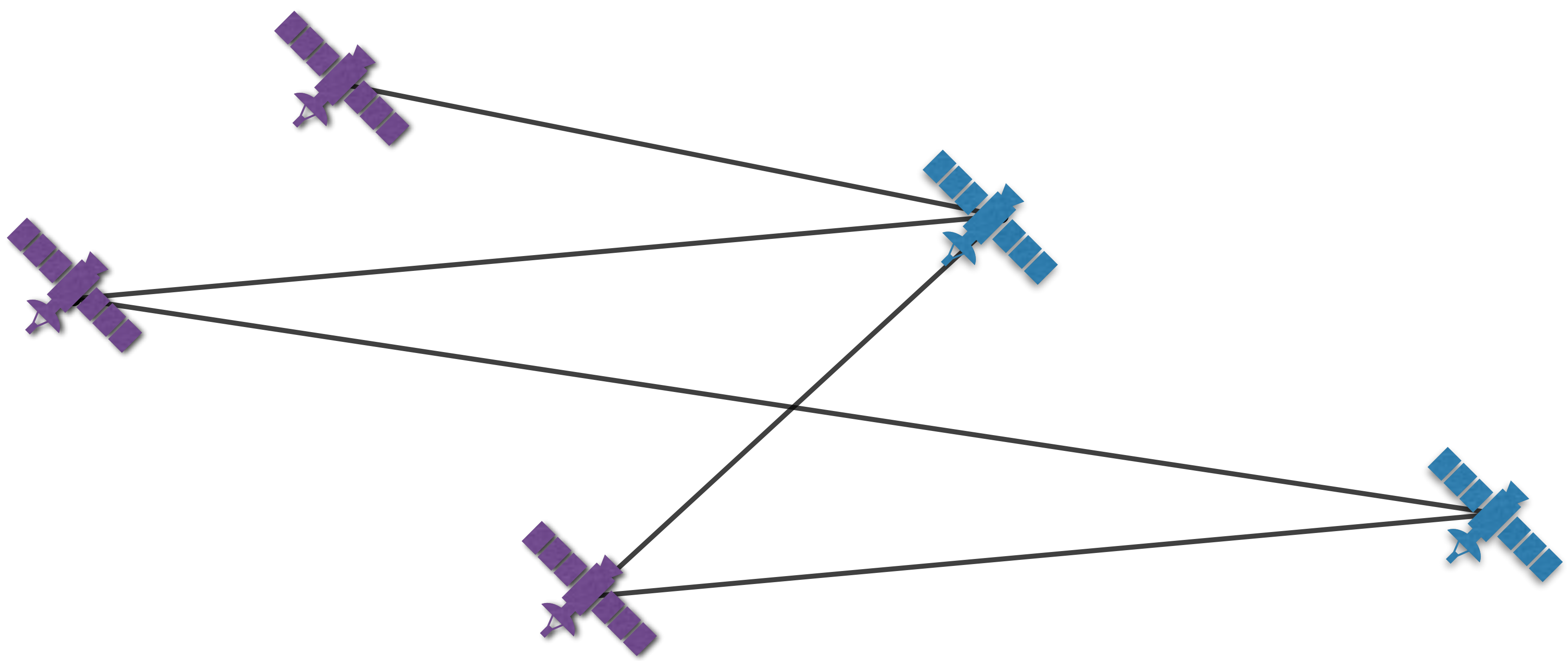


$$\max_{s \in V} \text{ optimal solution of } s \text{ to cover neighbors}$$

i.e., the largest smallest cone that encloses all neighbors

Approximation for bipartite graphs

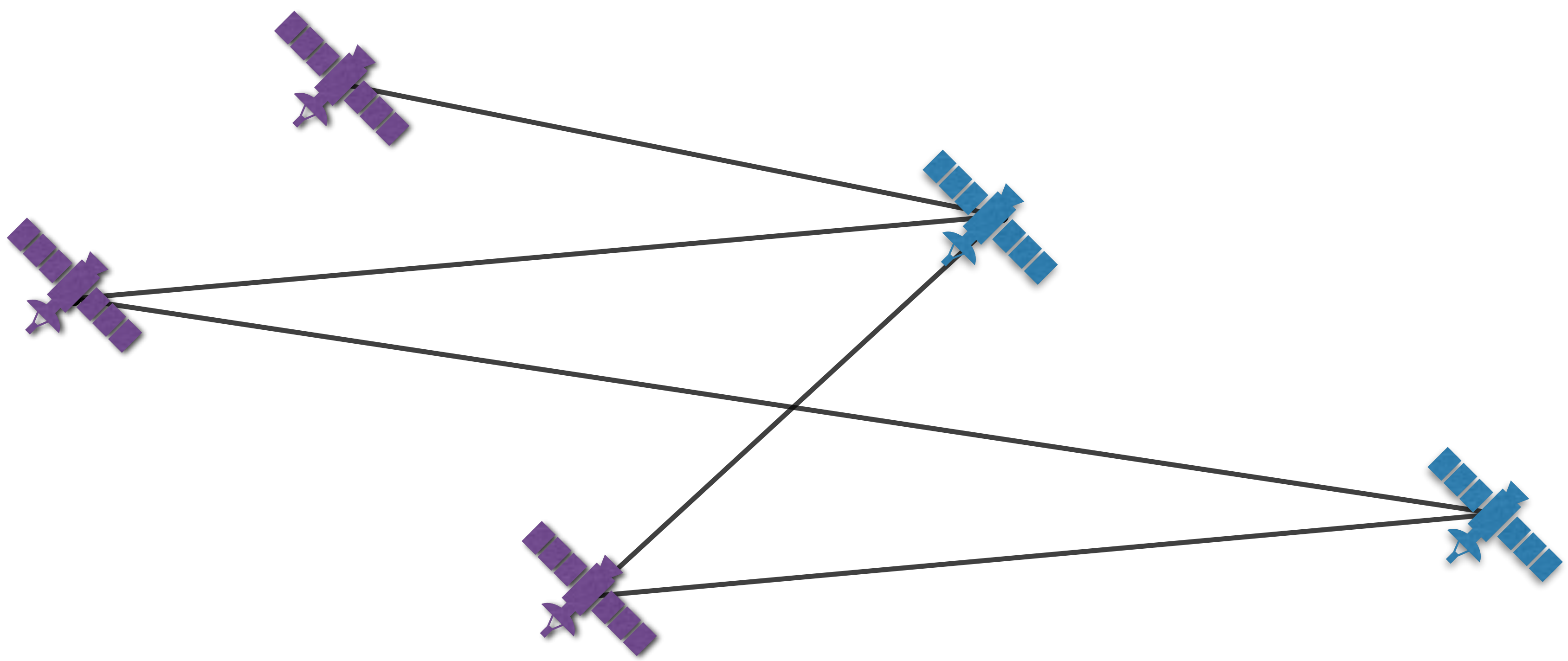
▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*



Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy



Sándor Fekete, Linda Kleist, and Dominik Krupke

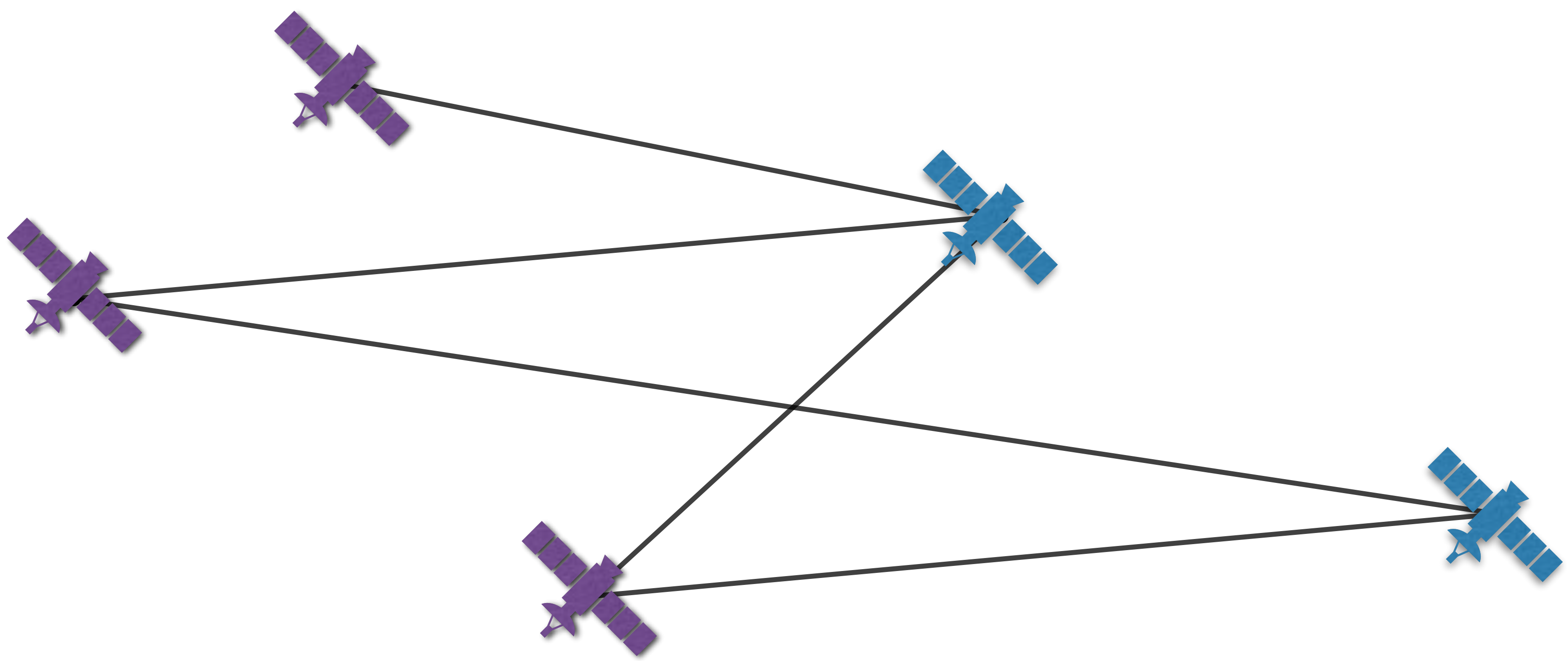
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:

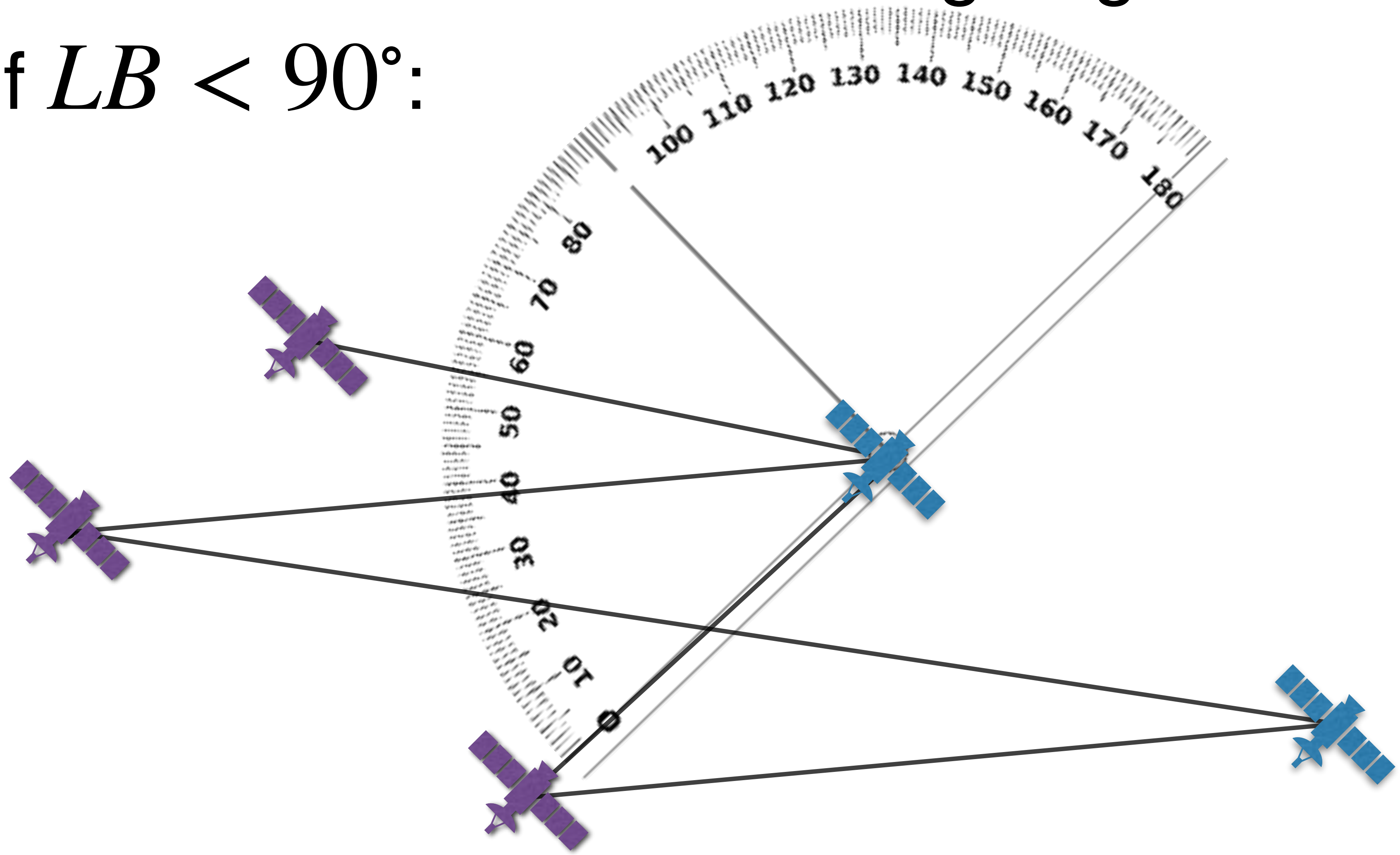


Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:

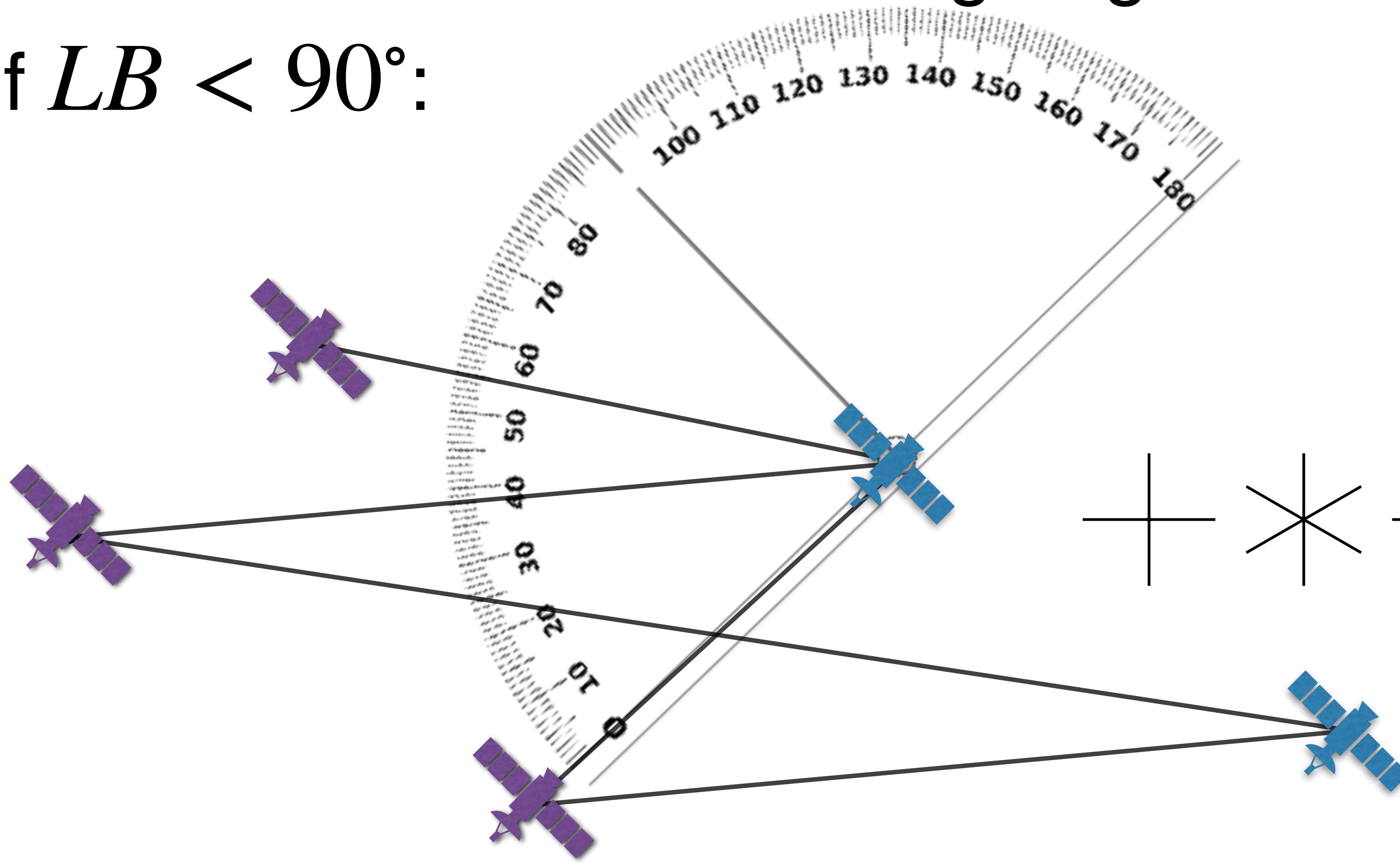


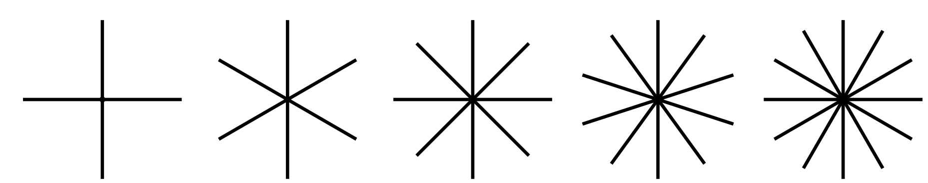
Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90^\circ$: Use alternating angle strategy

If $LB < 90^\circ$:





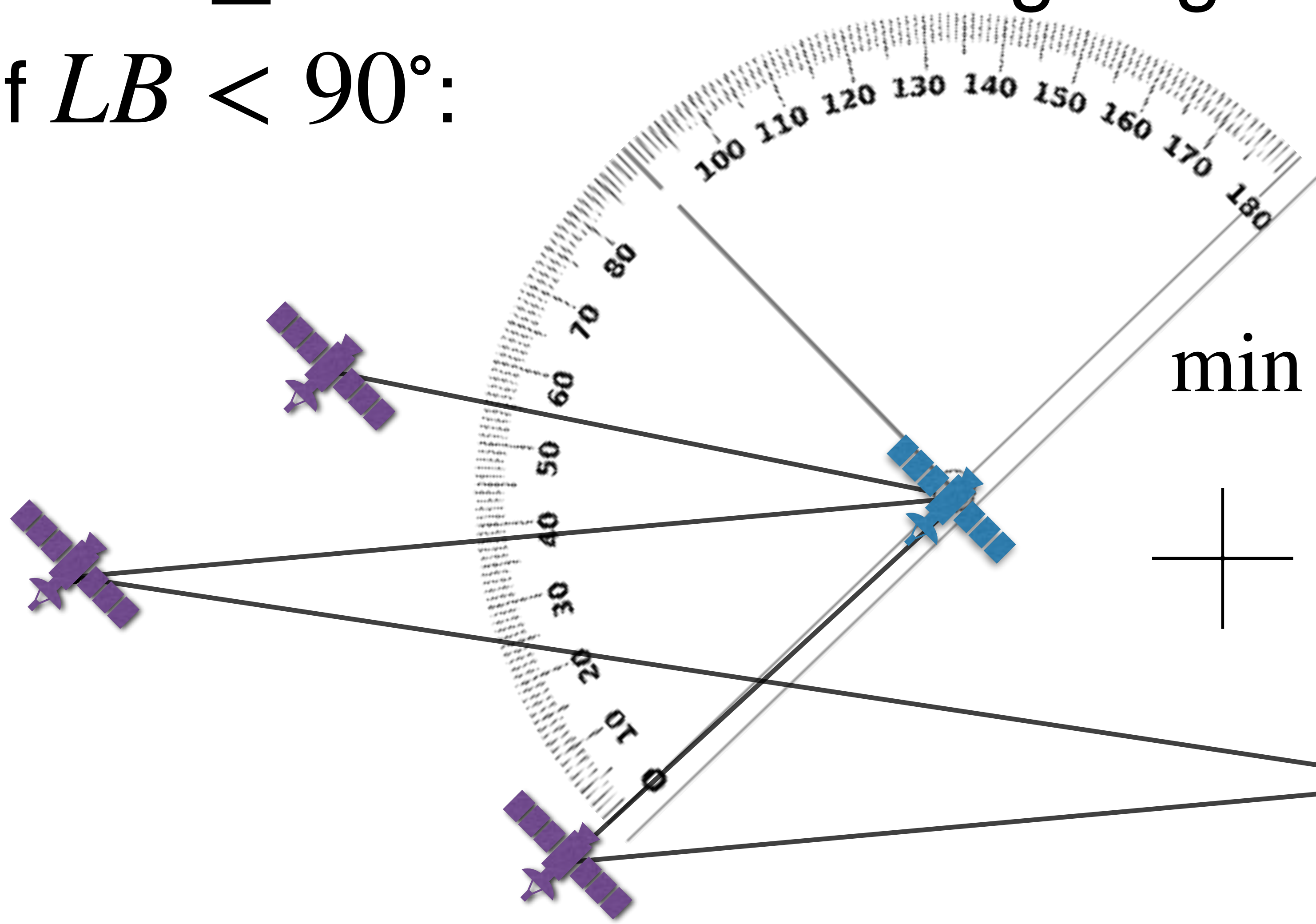
d.krupke@tu-bs.de

Approximation for bipartite graphs

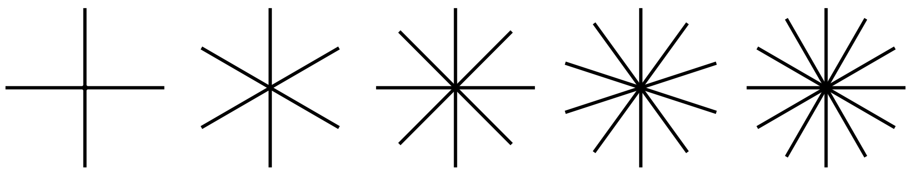
▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$



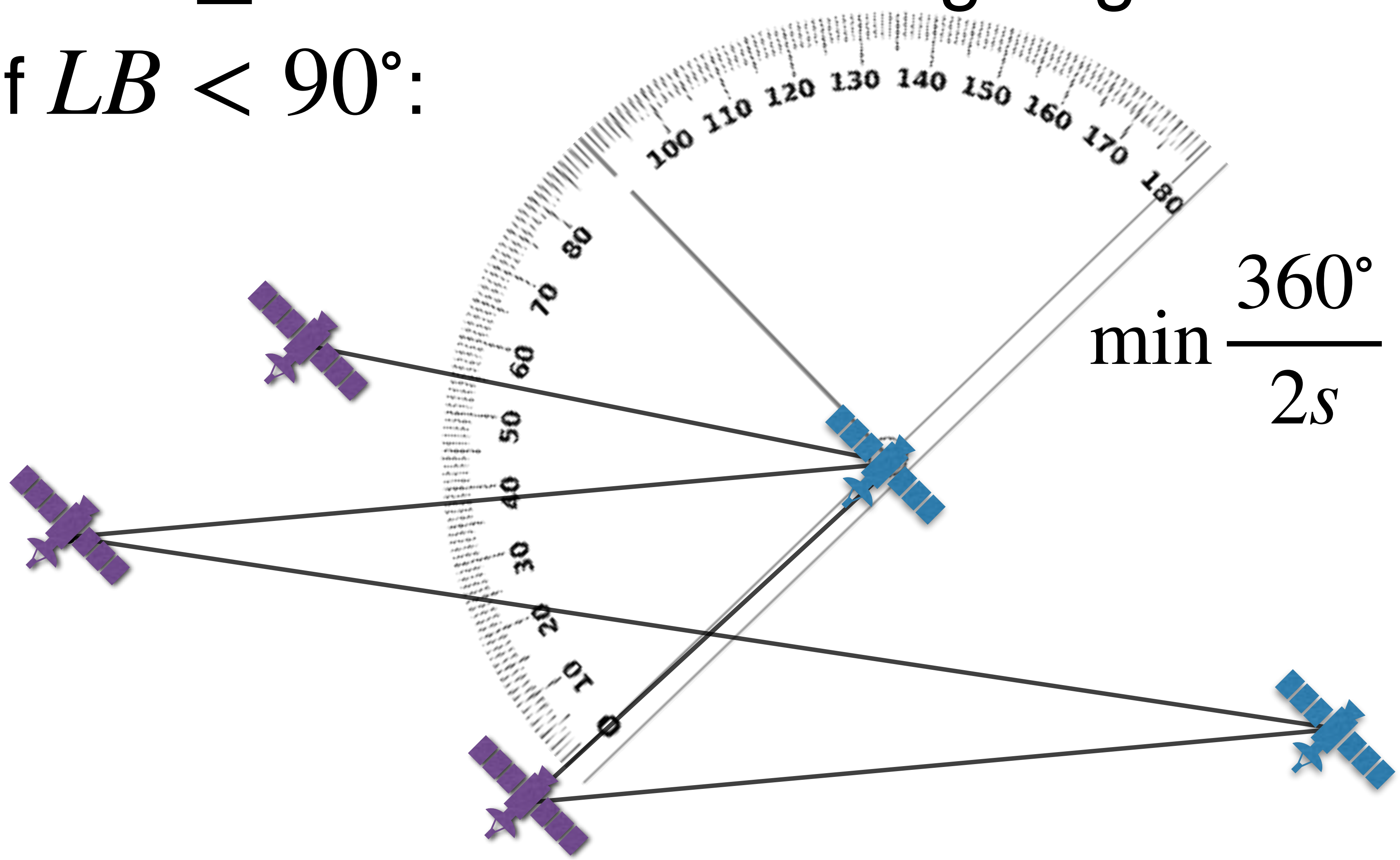
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$

d.krupke@tu-bs.de

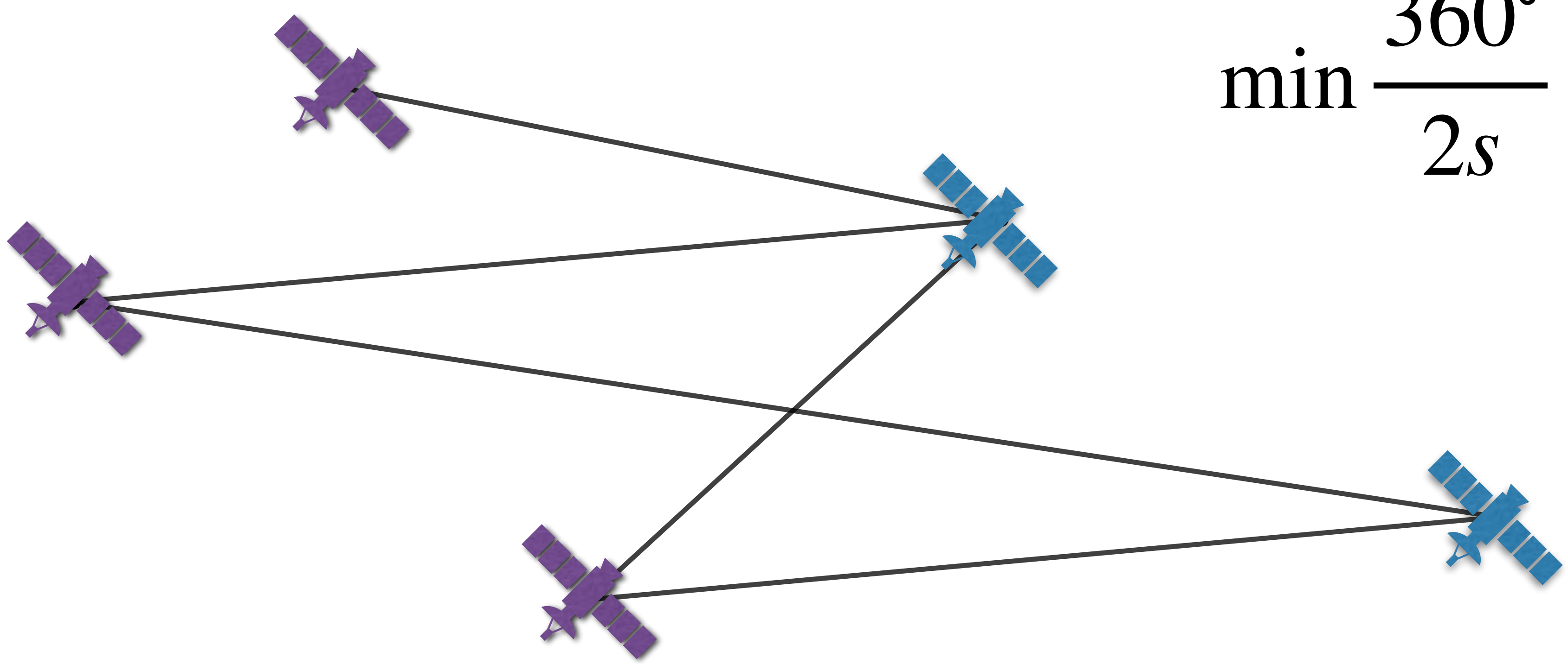
Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:

$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$



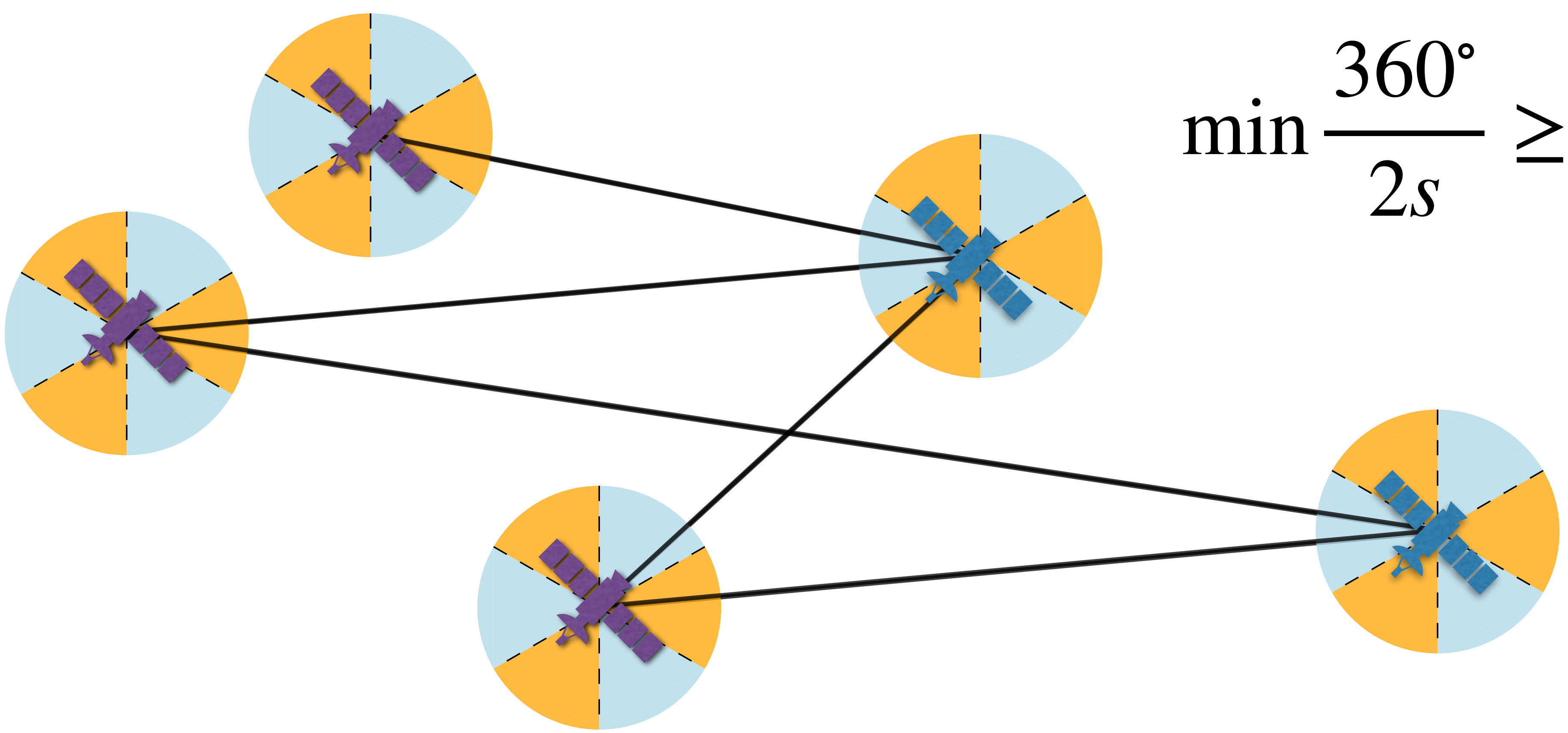
Technische Universität Braunschweig
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$

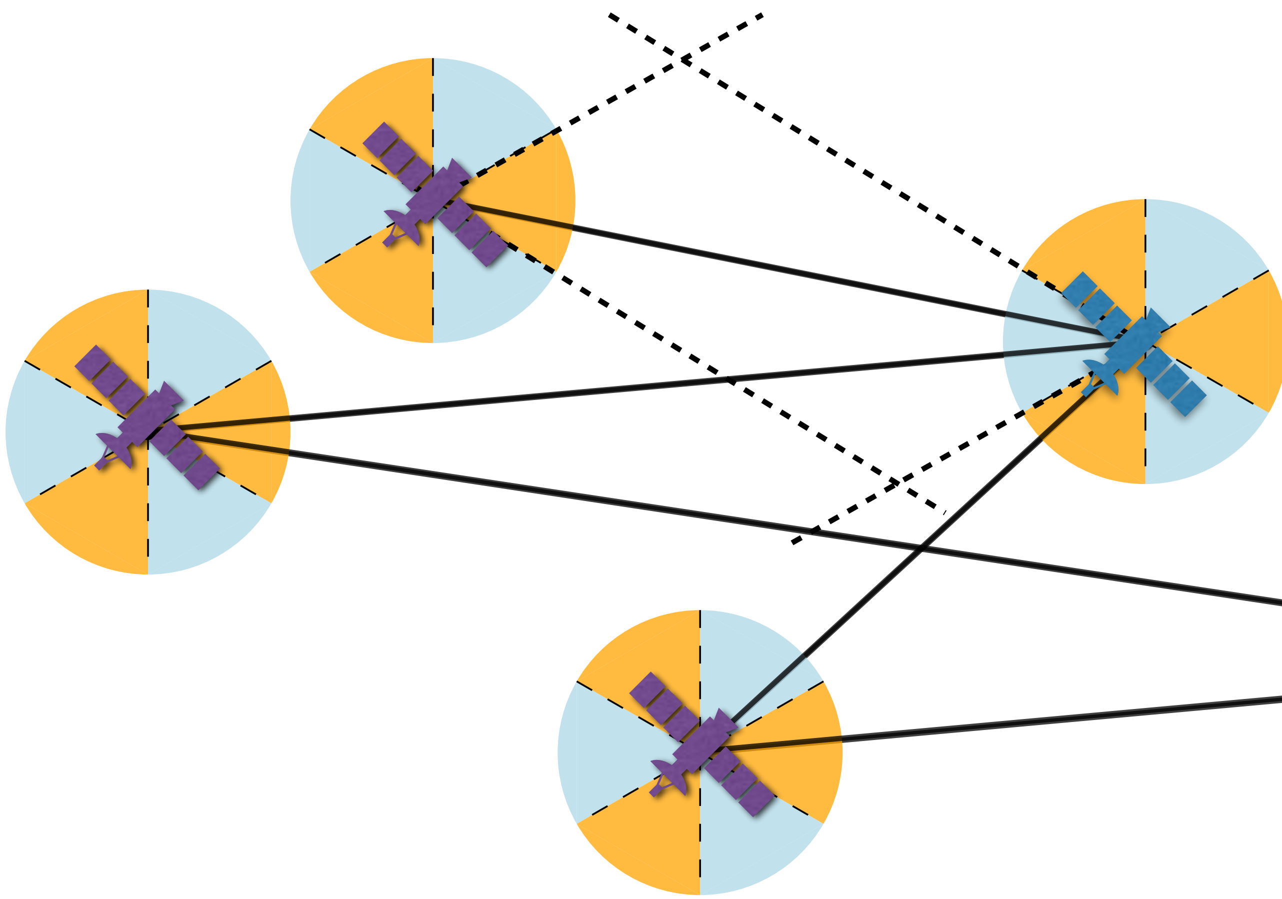
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$

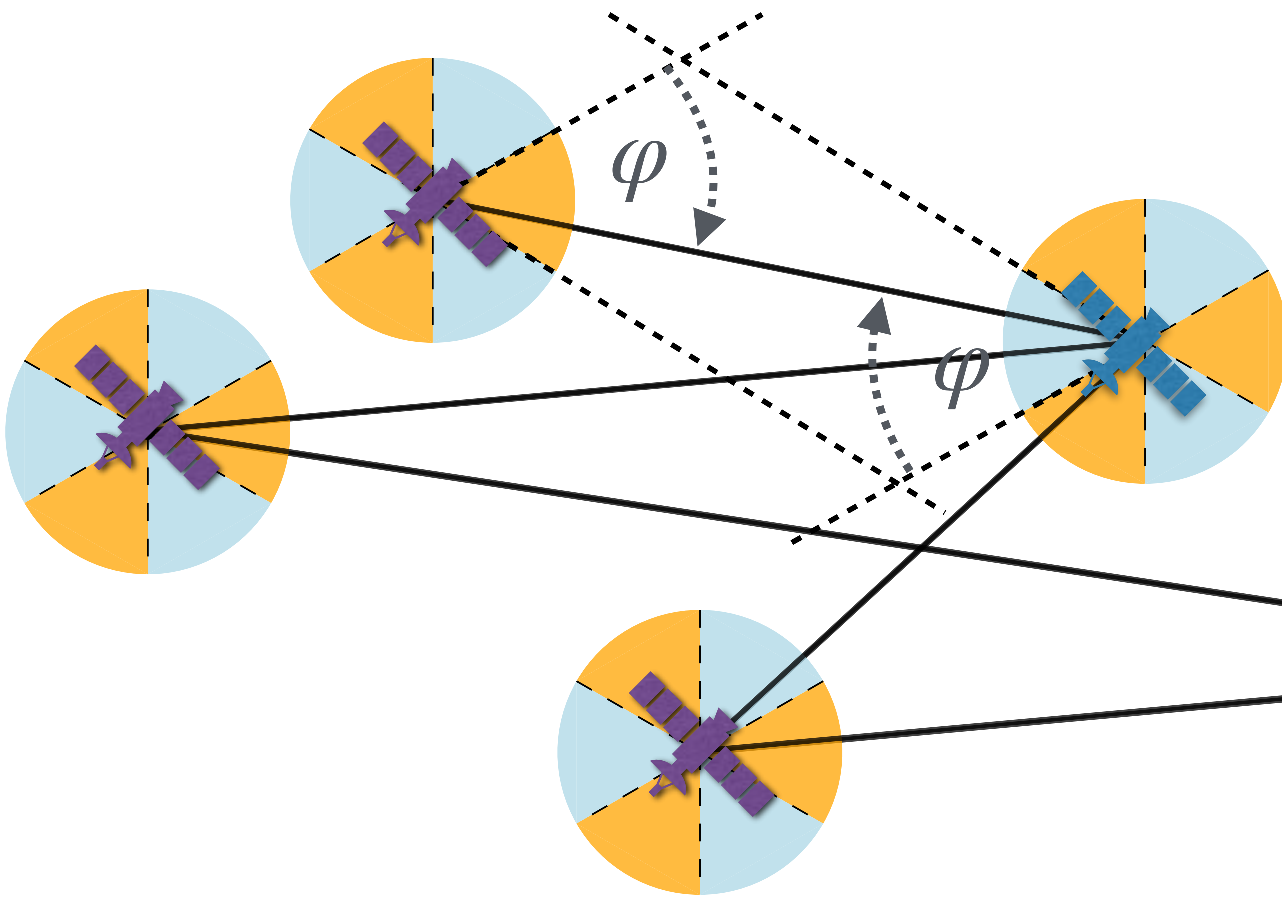
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$

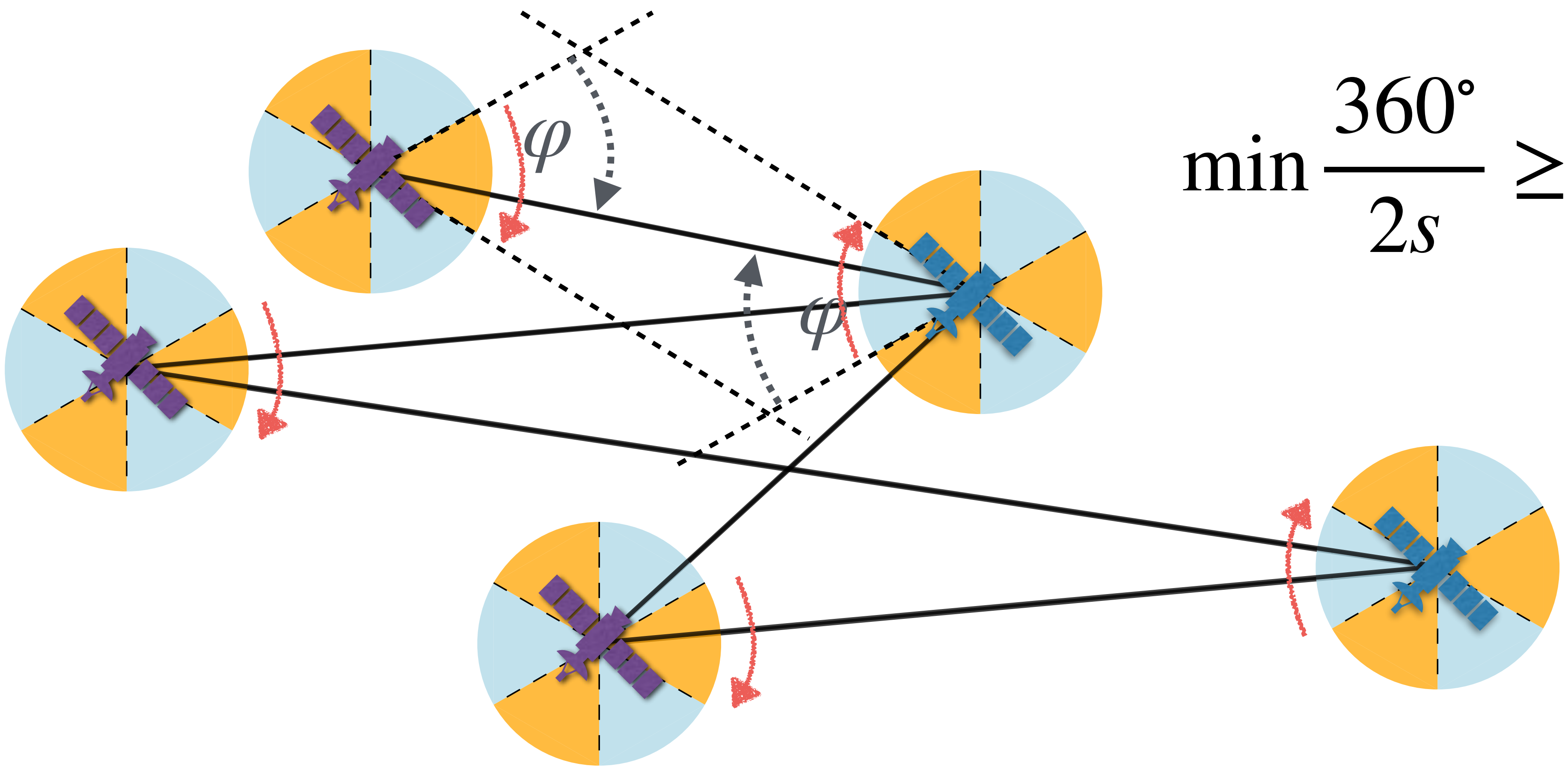
Technische Universität Braunschweig
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$

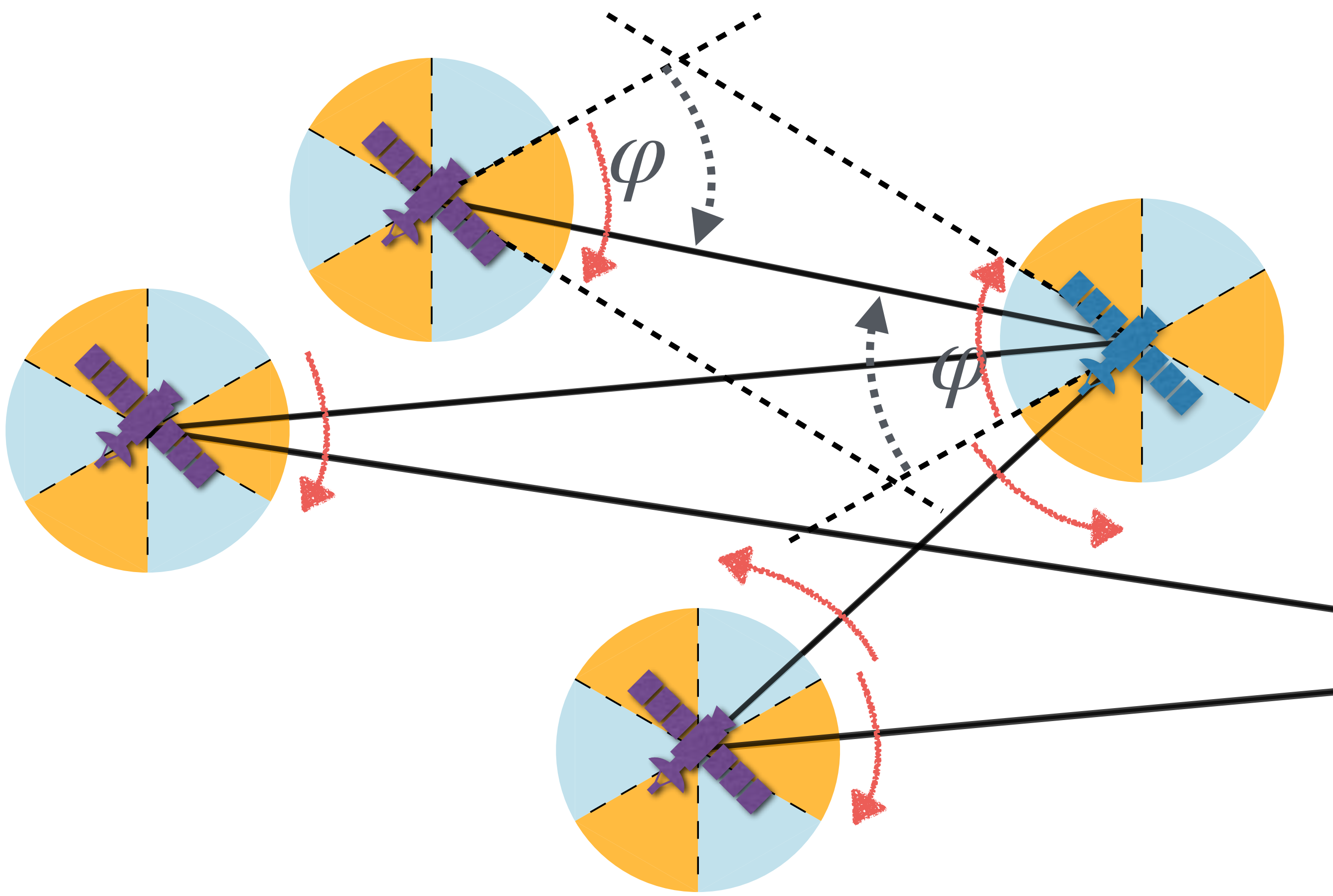
d.krupke@tu-bs.de

Approximation for bipartite graphs

▶ **Theorem 6.** *There is a 4.5-approximation algorithm for* MSC *for bipartite graphs in 2D.*

If $LB \geq 90°$: Use alternating angle strategy

If $LB < 90°$:



$$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$$

d.krupke@tu-bs.de

Chromatic Number and Approximations

▶ **Lemma 7.** *Every instance I of* MSC *in $\mathbb{R}^d$ needs a scan time T of at least $\Omega(\log_2 \chi(G_I))$, with G_I denoting the underlying graph of I. More precisely, $T \geq \frac{\lceil \log_2 \chi(G) \rceil - d}{d} \cdot 90°$.*

d.krupke@tu-bs.de

Chromatic Number and Approximations

▶ **Lemma 7.** *Every instance I of* MSC *in $\mathbb{R}^d$ needs a scan time T of at least $\Omega(\log_2 \chi(G_I))$, with G_I denoting the underlying graph of I. More precisely, $T \geq \frac{\lceil \log_2 \chi(G) \rceil - d}{d} \cdot 90^\circ$.*

Proof idea: With 90°, only 2^d-partite subgraphs can be scanned

Chromatic Number and Approximations

▶ **Lemma 7.** *Every instance I of* MSC *in $\mathbb{R}^d$ needs a scan time T of at least $\Omega(\log_2 \chi(G_I))$, with G_I denoting the underlying graph of I. More precisely, $T \geq \frac{\lceil \log_2 \chi(G) \rceil - d}{d} \cdot 90°$.*

Proof idea: With $90°$, only 2^d-partite subgraphs can be scanned

▶ **Corollary 8.** MSC *in 2D allows the following approximation factors.*

1. $O(\log_2 n)$ *for all graphs. Furthermore, the minimum scan time lies in $\Theta(\log_2 \chi(G))$.*
2. $O(1)$ *for planar graphs.*
3. $O(\log_2 d)$ *for d-degenerate graphs.*
4. $O(1)$ *for graphs of bounded treewidth.*
5. $O(1)$ *for complete graphs.*

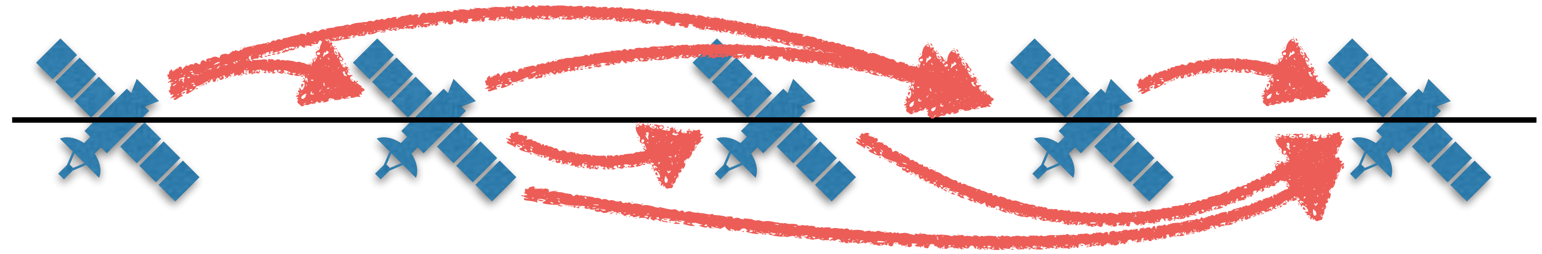
Summary

Summary

1D

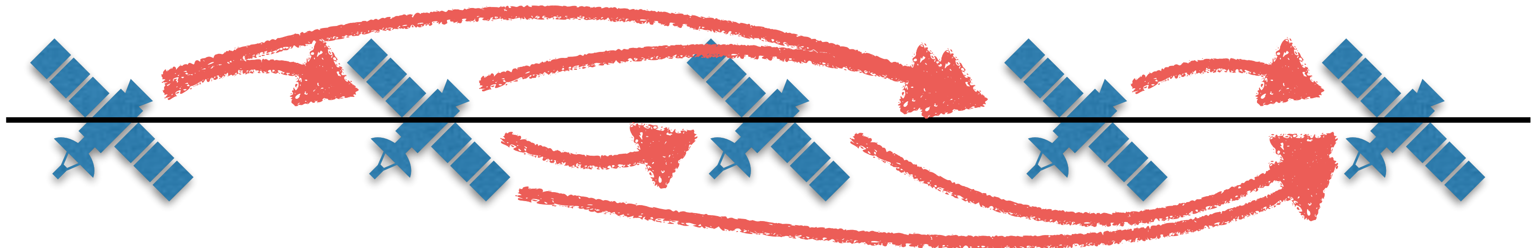
Summary

1D



Summary

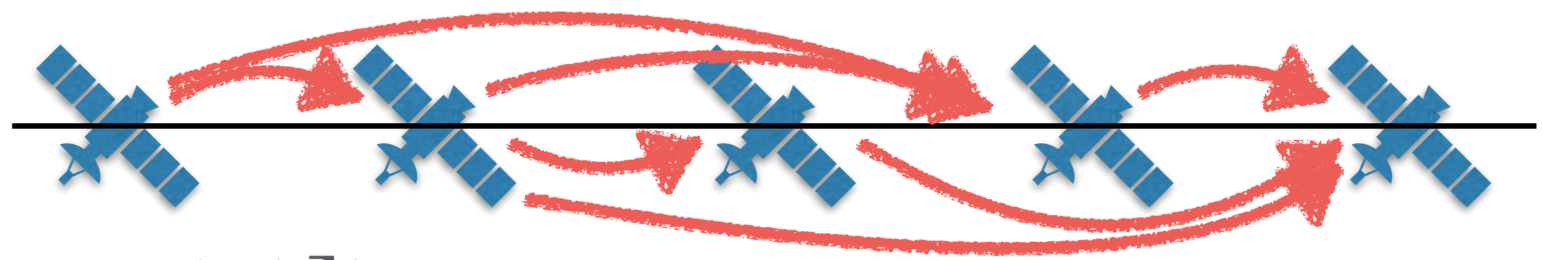
1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

Summary

1D

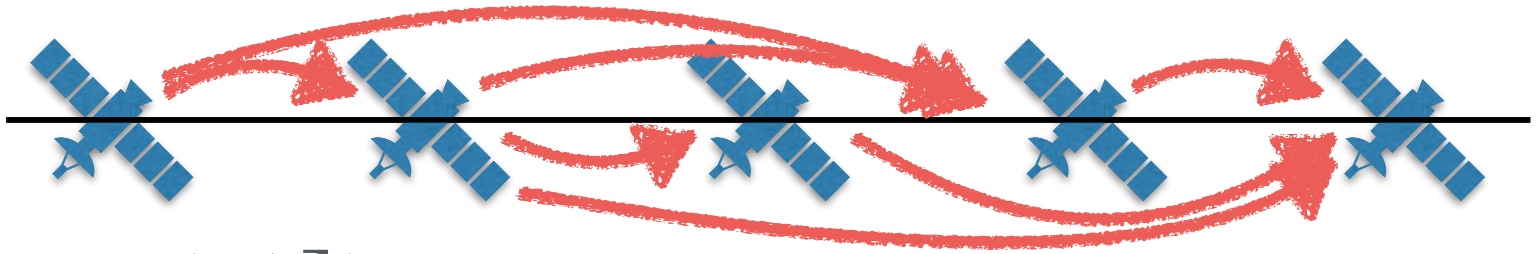


$\Theta(\lceil \log_2 \chi(G) \rceil)$

d.krupke@tu-bs.de

Summary

1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

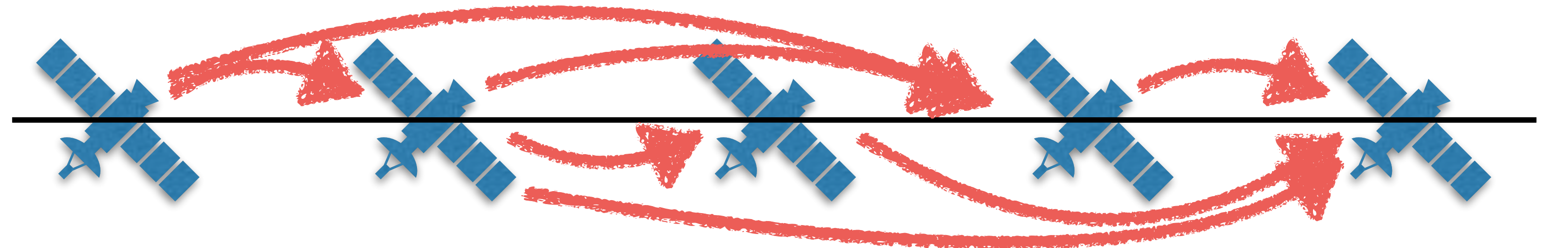
2D

d.krupke@tu-bs.de

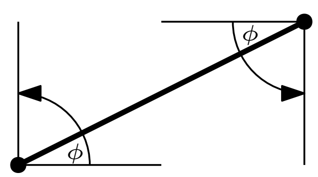
Summary

1D

$\Theta(\lceil \log_2 \chi(G) \rceil)$



2D

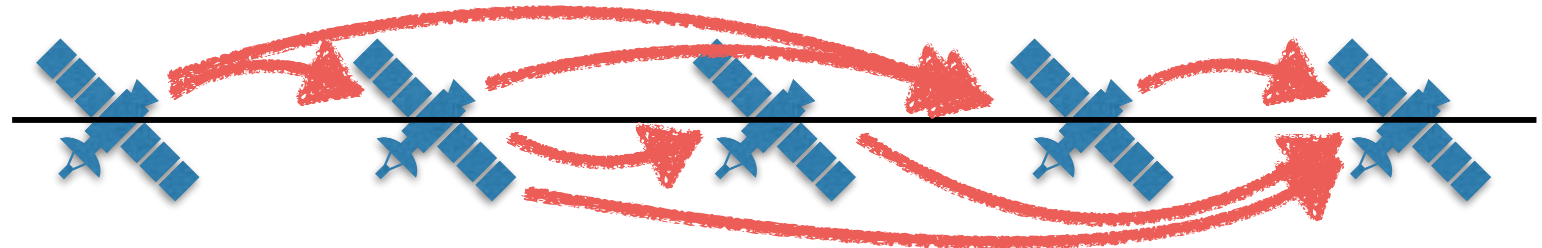


Sándor Fekete, Linda Kleist, and Dominik Krupke

d.krupke@tu-bs.de

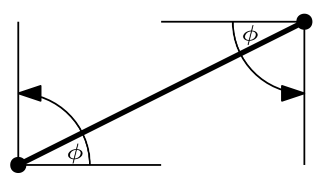
Summary

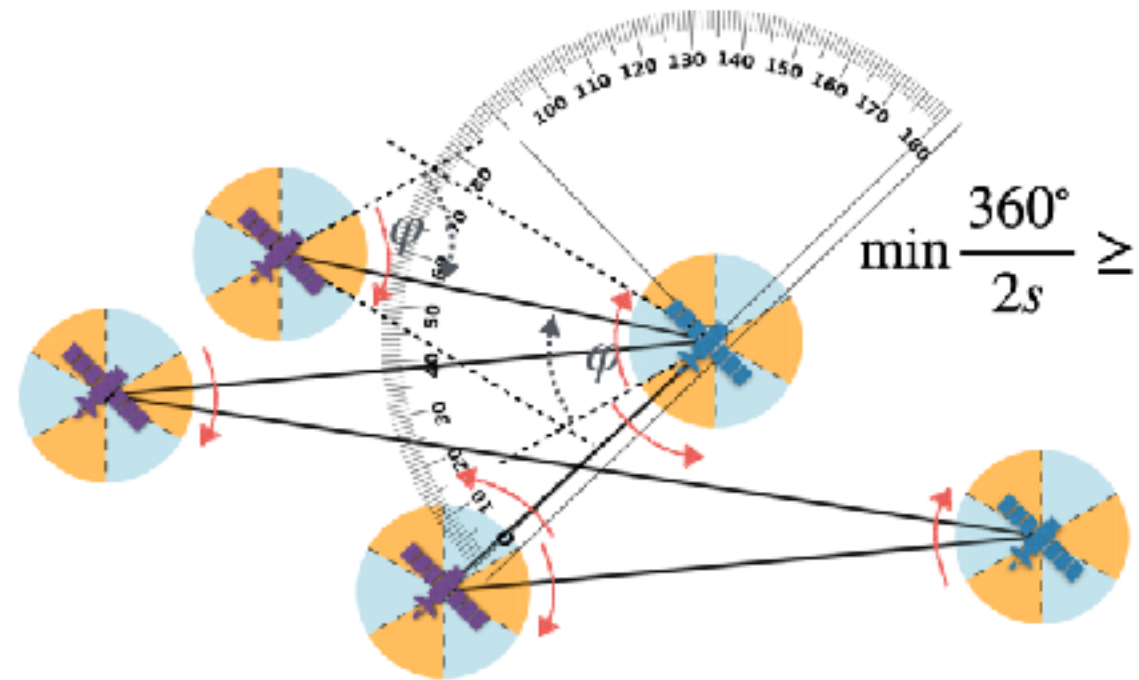
1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

2D

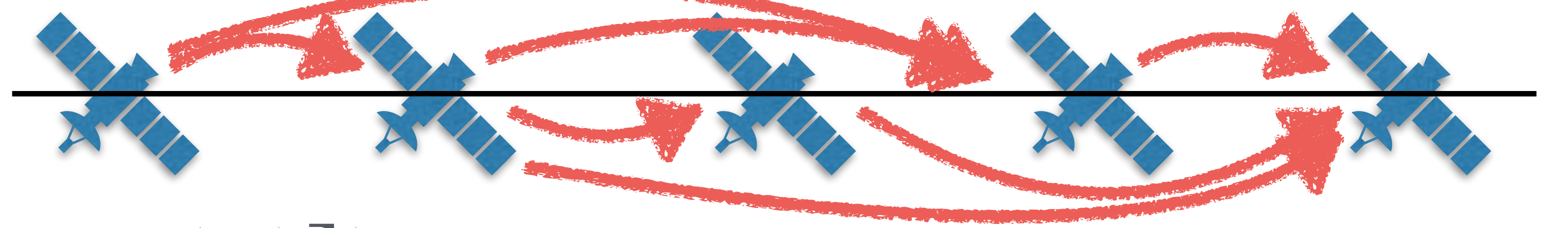




$$\min \frac{360^\circ}{2s} \geq 55^\circ \Rightarrow 60^\circ$$

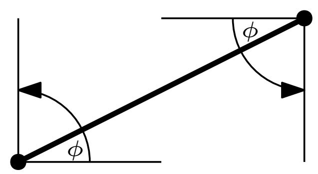
Summary

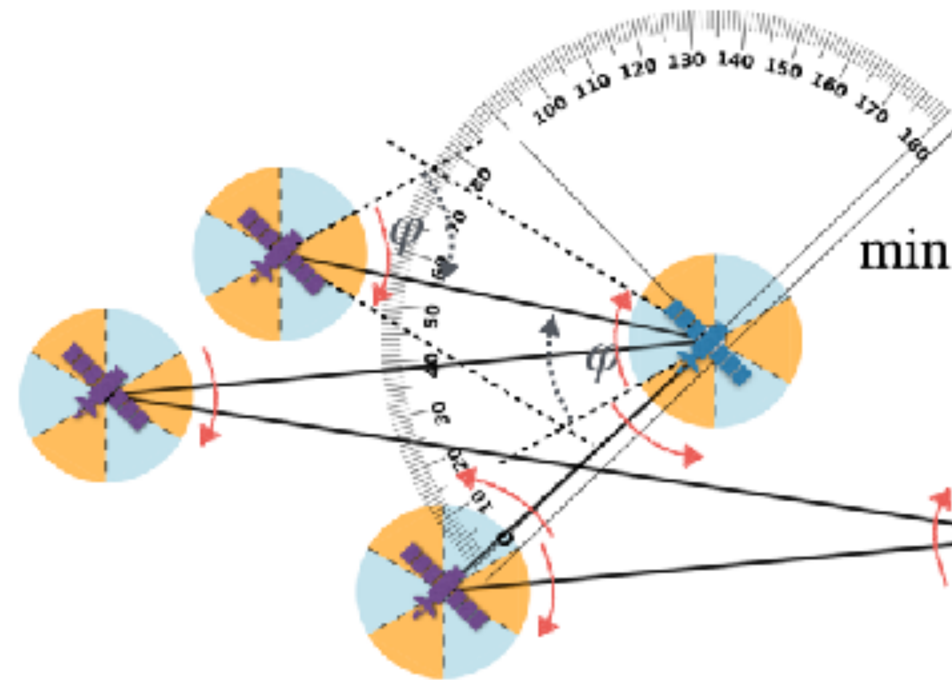
1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

2D





$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$

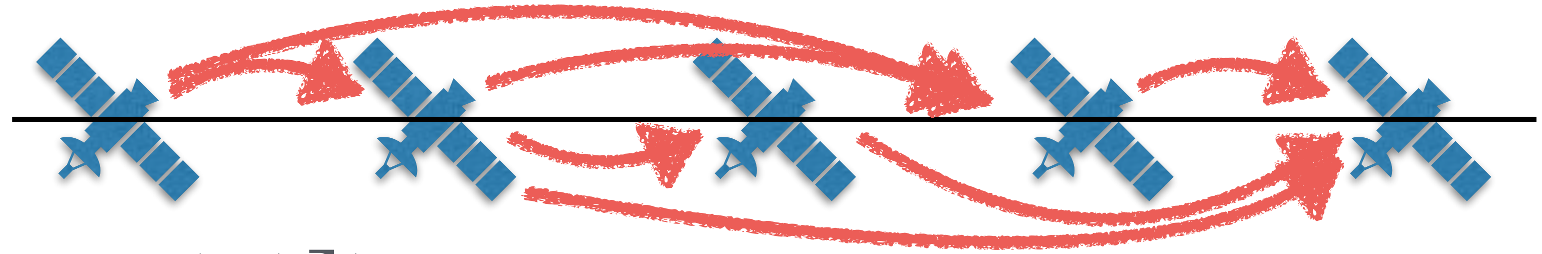
$\Theta(\lceil \log_2 \chi(G) \rceil)$

Sándor Fekete, Linda Kleist, and Dominik Krupke

d.krupke@tu-bs.de

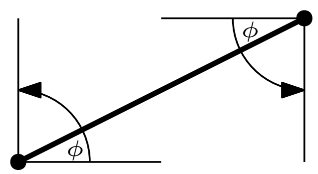
Summary

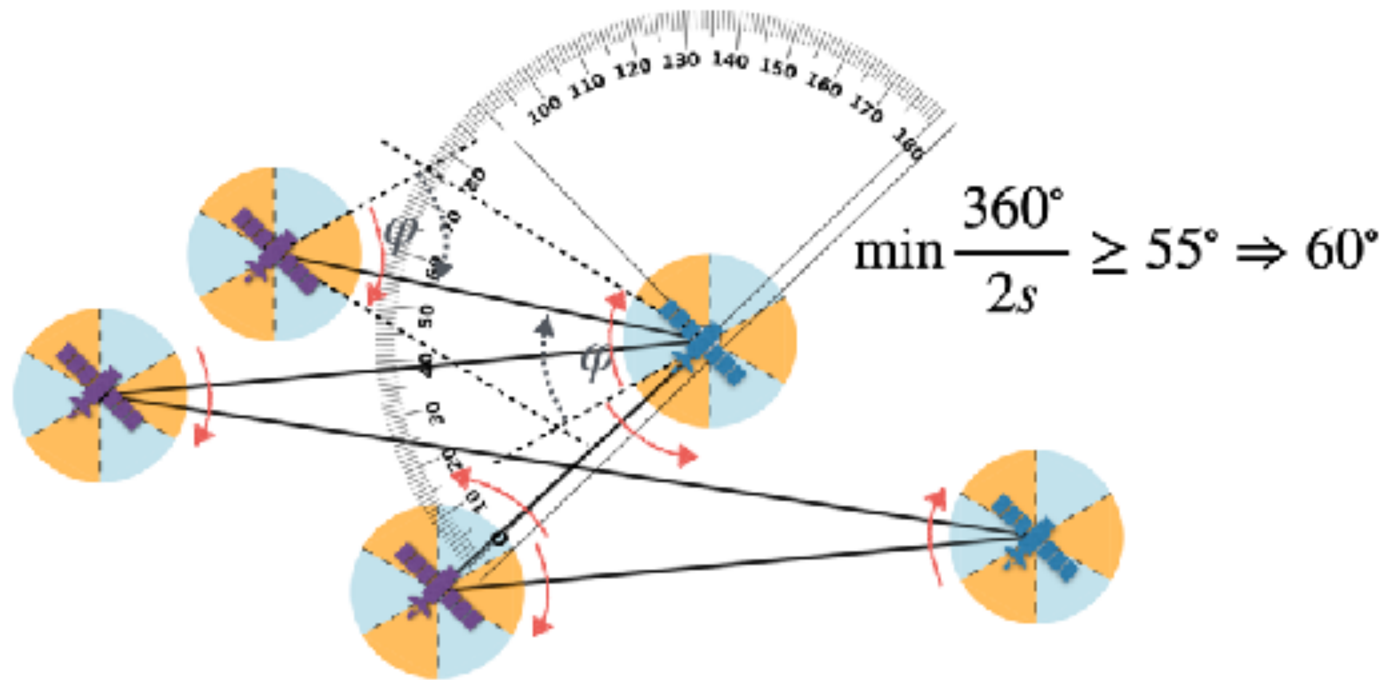
1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

2D

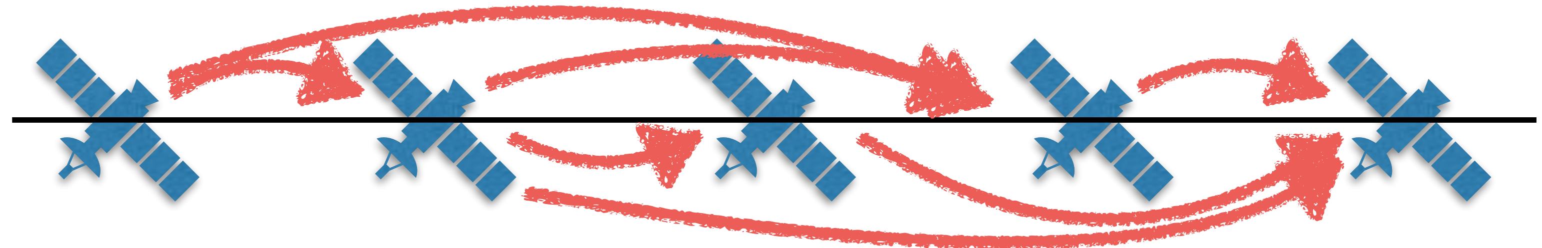




$\Theta(\lceil \log_2 \chi(G) \rceil)$

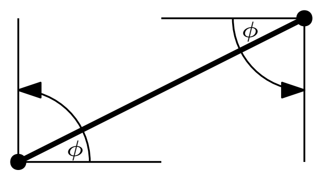
Summary

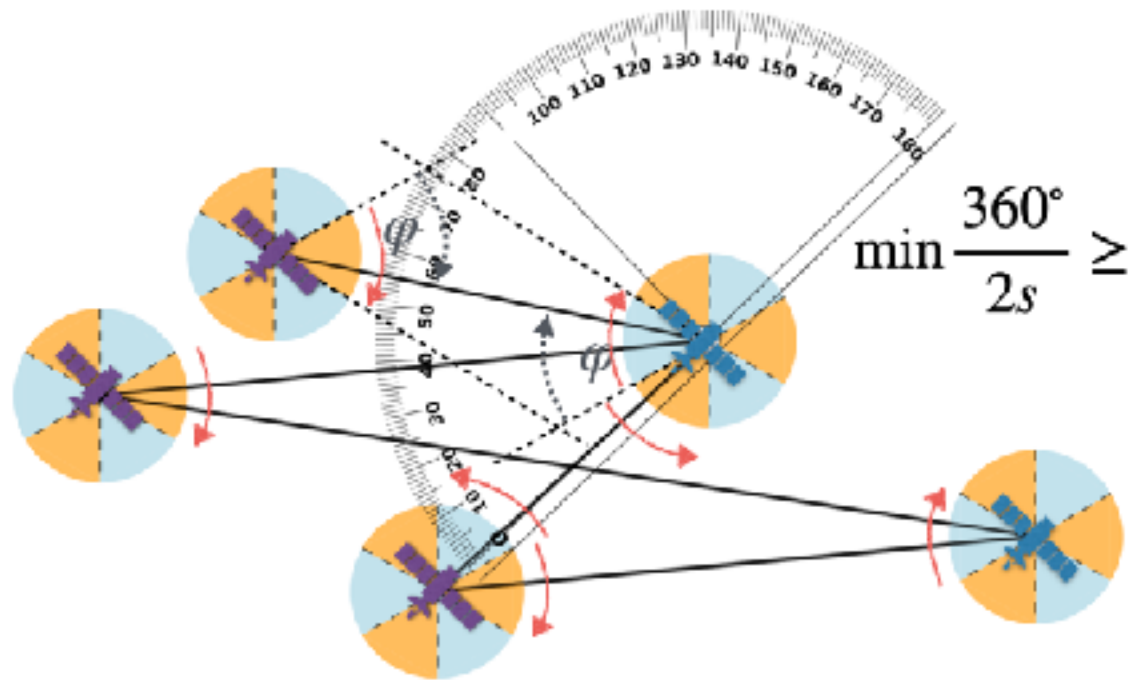
1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

2D





$\min \frac{360°}{2s} \geq 55° \Rightarrow 60°$

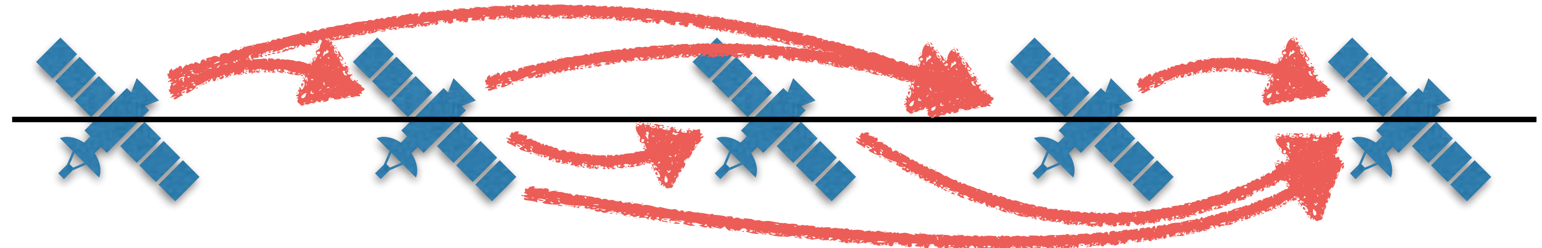
$\Theta(\lceil \log_2 \chi(G) \rceil)$

3D & Abstract

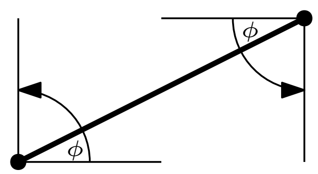
Summary

1D

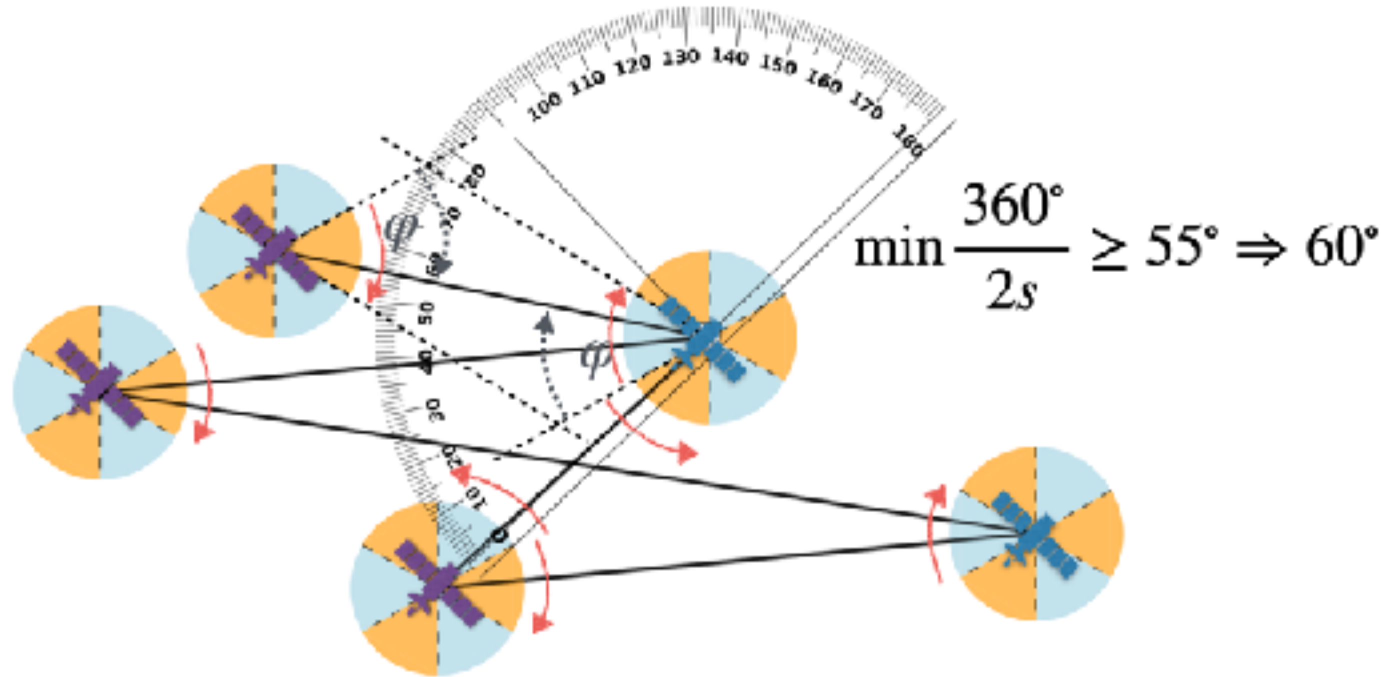
$\Theta(\lceil \log_2 \chi(G) \rceil)$



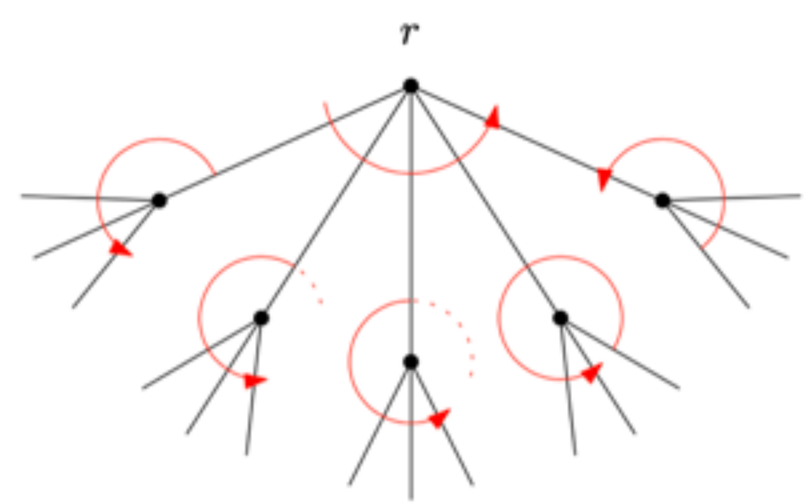
2D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

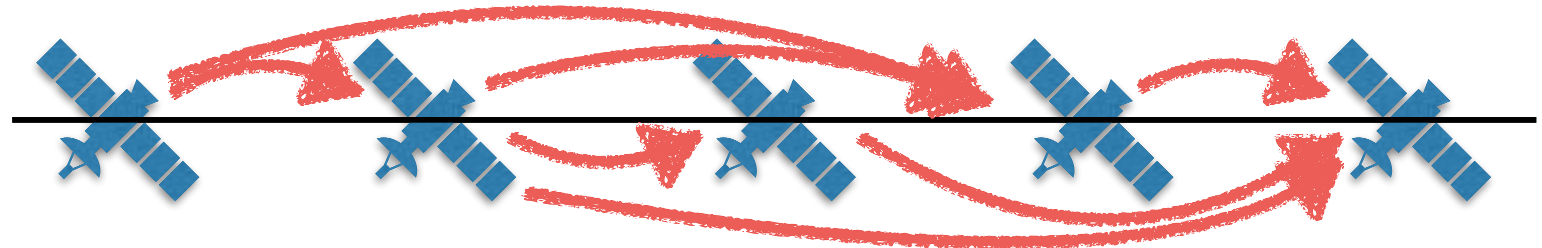


3D & Abstract



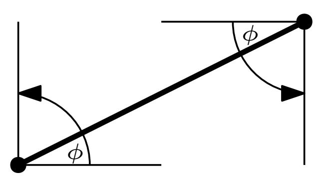
Summary

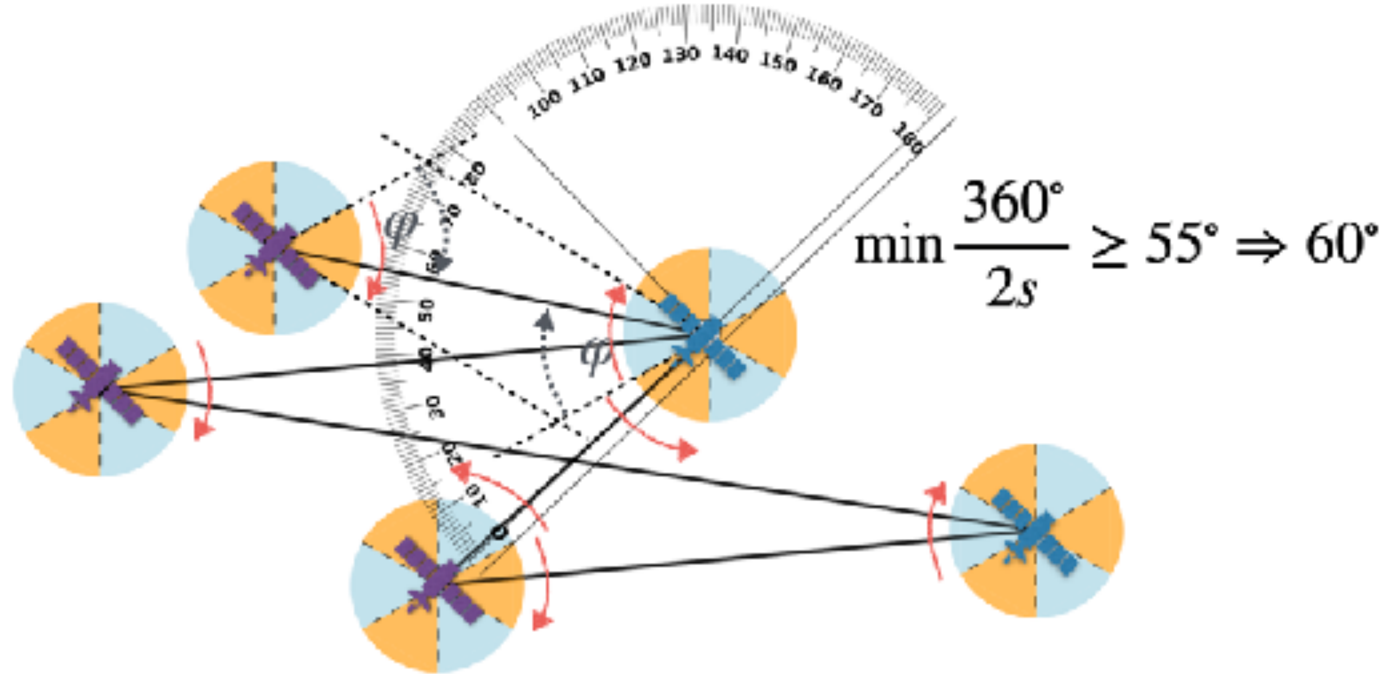
1D



$\Theta(\lceil \log_2 \chi(G) \rceil)$

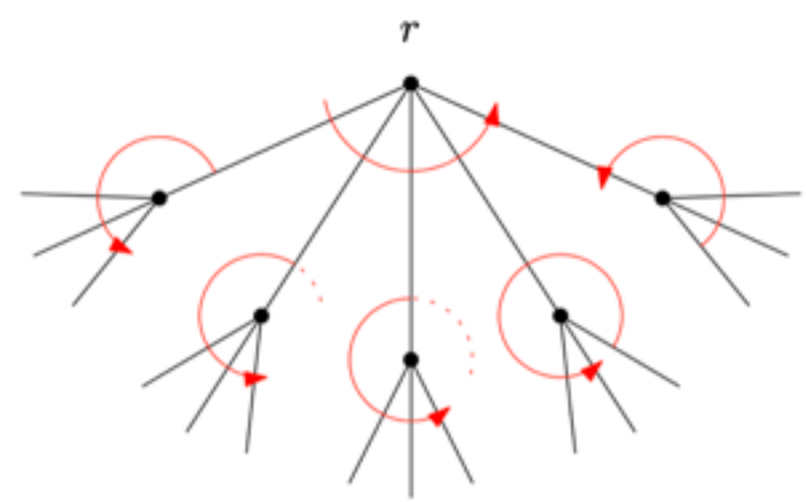
2D





$\Theta(\lceil \log_2 \chi(G) \rceil)$

3D & Abstract



$\Omega(\lceil \log_2 \chi(G) \rceil)$

d.krupke@tu-bs.de

Thank you!