

On Restricting Separator Problems in the OBLOT Computational Landscape

Quentin BRAMAS and Sébastien TIXEUIL

University of Strasbourg, Sorbonne Université, LIP6, IUF

NETYS 2025

Slides and paper available at https://bramas.fr

Université

de Strasbourg



















Can solve task T

Under hypothesis H





Model, Task

Solvable

 $\{H_1, H_2, H_3\}$ $\{H_1, H_2\}$

Unsolvable

 $\{H_1, H_3\}$





Model



Task

Task H1 H2 H3







Model.



Task $\notin \{H_1, H_2\} \{H_1, H_3\}$

Task E H1 H2 H3





Model



Landscape





Model



Landscape

















11











































n1: LCM

Configmation 1 => Configmation 2





 $\mathcal{R}_1: LCM$ $\mathcal{R}_2: LCM$ $\subset Migmation 1 \xrightarrow{\mathcal{R}_2} \dots$





























FSYNC: Fully-Synchronous

.

















٢4 C_{3} , LCM CM/ |

. -

. .

/









SSYNC: Semi-Synchronous









63

. . . .





ASYNL









Ø

Algo: mid 0---->0





- Start at the origin (0, 0).

- Move 1 unit to the right, then 1 unit up, then 2 units to the left, then 2 units down, t the right, and so on, increasing the step size by 1 after every two turns. Qwen3-30b6:





Mistral:

- Start at - Move 1 the right, a Qwen3-3

Thought Process

Okay, let's try to figure out how to approach this problem. So, there are two dimensionless robots, me and another one. We're both executing the same algorithm, and the goal is to gather at the same point. But I don't know if the other robot is currently activated like I am. Hmm, what should I do?

First, I need to understand the problem better. Since both robots are dimensionless, maybe they don't have any physical size, which might affect how they move or interact. But the main issue is that we need to gather at the same point, but we don't know the other robot's status. So, maybe the challenge is about coordination without knowing if the other is active or not.

Since we're both executing the same algorithm, maybe there's a standard protocol for gathering. But the problem is the uncertainty about the other's activation. So, perhaps the solution involves some kind of synchronization or signal that allows us to determine if the other is active and then move towards them.

Wait, but how can we communicate if we don't have any sensors or communication methods? The problem doesn't mention any. So maybe we have to rely on movement patterns or some prearranged strategy.

If we both follow the same algorithm, maybe we need to decide on a strategy that ensures we meet regardless of activation status. For example, if we both move in a certain pattern, like spiraling out or moving in a circle, but that might not work if one is inactive.

Alternatively, since we can't communicate, maybe we need to use some form of randomness. Like, each robot has a chance to move in a certain direction or stay still, but that might not guarantee meeting.

https://bramas.fr

then 2 units down, t turns.

~





- Start at the origin (0, 0). - Move 1 unit to the right, then 1 unit up, then 2 units to the left, then 2 units down, t the right, and so on, increasing the step size by 1 after every two turns. <u>Owen3-30h6</u> • Two dimensionless robots.

Thought Process

Okay, let's try to figure out how to approach this problem. So, there are two dimensio me and another one. We're both executing the same algorithm, and the goal is to gat same point. But I don't know if the other robot is currently activated like I am. Hmm, v do?

First, I need to understand the problem better. Since both robots are dimensionless, r don't have any physical size, which might affect how they move or interact. But the m that we need to gather at the same point, but we don't know the other robot's status. challenge is about coordination without knowing if the other is active or not.

Since we're both executing the same algorithm, maybe there's a standard protocol fo But the problem is the uncertainty about the other's activation. So, perhaps the soluti some kind of synchronization or signal that allows us to determine if the other is activ move towards them.

Wait, but how can we communicate if we don't have any sensors or communication n problem doesn't mention any. So maybe we have to rely on movement patterns or so prearranged strategy.

If we both follow the same algorithm, maybe we need to decide on a strategy that en: regardless of activation status. For example, if we both move in a certain pattern, like or moving in a circle, but that might not work if one is inactive.

Alternatively, since we can't communicate, maybe we need to use some form of rand each robot has a chance to move in a certain direction or stay still, but that might not meeting.

But the problem says we're executing the same algorithm. So maybe the algorithm needs to account for the possibility that the other is inactive. So perhaps the algorithm includes a waiting period, or a way to detect the other robot's presence.

Wait, but since we're dimensionless, maybe we can't detect each other unless we are at the same point. So maybe the algorithm needs to have a way to move towards a point where we can meet, but how?

Wait, the problem says "you don't know if the other is activated right now like you are." So I'm Another thought: if both robots are in the same environment, maybe they can use the environment's activated, and I don't know if the other is activated. So maybe the other could be inactive or active. features. But since the problem doesn't specify the environment, maybe it's a 2D plane or But the algorithm is the same for both. So the algorithm needs to work whether the other is active something. or not.

In standard robot gathering problems, sometimes robots use a strategy like moving towards a common point, or using a symmetry-breaking approach. But since we don't know if the other is active, maybe we need to assume that if the other is active, they're also following the same algorithm. So maybe the algorithm is designed to have both robots converge to a single point regardless of their initial positions.

But how do they determine that point? If they can't communicate, maybe they use a random point or a fixed reference. However, if both choose the same point, they'll meet. But if they choose different points, they won't. So maybe the algorithm includes a way to decide on a common reference.

Wait, but if I'm the only one active, then the other is inactive. So maybe the algorithm includes a way to wait for the other to activate, but if the other is never activated, then I might just keep waiting. Alternatively, if the robots can sense each other's presence, but the problem doesn't mention that. But that's not efficient. So perhaps the problem assumes that the robots can only move and not sense anything else.

Alternatively, maybe the algorithm is designed so that if the other is active, we both move towards Wait, but if they can't sense each other, how can they know when they meet? Maybe the problem is each other, and if the other is inactive, I stay still. But without knowing the other's status, how can I more about the algorithm's design that ensures that if both are active, they meet, but if one is inactive, the active one can still gather. But the problem says both are executing the same decide? algorithm, so maybe the algorithm needs to handle both cases. This seems like a problem of coordination without communication. In distributed algorithms, there

This is a bit confusing. Let me think again. The key points:

https://bramas.fr

Same algorithm.

- Goal is to gather at the same point.
- Don't know if the other is activated (i.e., maybe one is inactive, the other is active, or both are active? Or maybe the other's activation status is unknown, but both are executing the same algorithm, which might mean they're both active? Or maybe the algorithm is such that the other might not be activated yet.)

So, if the other is active, then both are executing the algorithm. If the other is inactive, then only I am executing it. So the algorithm needs to handle both scenarios.

In the case where the other is inactive, I need to find a way to gather, but since the other isn't moving, maybe I just stay still? But if the other is active, then I need to move towards them.

But how can I know if the other is active or not? The problem states that I don't know. So the algorithm must be designed in a way that regardless of the other's activation status, the gathering is achieved.

are strategies for such problems. One common approach is to use a randomized algorithm where each robot has a certain probability to move or stay, which increases the chance of meeting. But since the robots are executing the same algorithm, maybe they both use the same strategy





- Start at the origin (0, 0). - Move 1 unit to the right, then 1 unit up, then 2 units to the left, then 2 units down, t the right, and so on, increasing the step size by 1 after every two turns. Qwen3-30b6: • Two dimensionless robots.

Thought Process

Okay, let's try to figure out how to approach this problem. So, t me and another one. We're both executing the same algorith same point. But I don't know if the other robot is currently acti do?

First, I need to understand the problem better. Since both robo don't have any physical size, which might affect how they move

Another idea: if both robots follow a deterministic algorithm that ensures they meet if both are active, but if only one is active, the active one can still gather. But how?

Alternatively, maybe the algorithm is to move in a specific pattern, like a spiral or a search pattern, that eventually covers all possible points. But without knowing the other's position, it's hard to say.

Wait, maybe the problem is similar to the "gathering problem" in distributed computing, where robots need to meet. In that case, common solutions involve symmetry breaking. For example, if the robots can choose a direction or move in a way that breaks symmetry.

But since both robots are executing the same algorithm, if they are in symmetric positions, they might move in the same way, leading to no progress. So the algorithm needs to have a way to break symmetry, perhaps by using some randomness or a unique identifier.

But the problem states that the robots are dimensionless, which might mean they don't have unique identifiers. So randomness might be the way to go.

So, the algorithm could be: each robot, with some probability, decides to move in a certain direction or stay still. Over time, the probability of them moving towards each other increases, leading to a meeting.

But how to formalize this? Maybe each robot has a random walk algorithm, moving in random directions. If both are active, they might eventually meet. If only one is active, it's unclear how that helps, but maybe the inactive one is considered as a fixed point, so the active one can move towards it. But without knowing where the other is, how?

Alternatively, maybe the algorithm is to move towards a predefined point, like the origin. But if both are moving towards the origin, they will meet there. However, if only one is active, then the active one goes to the origin, but the inactive one is already there? Or not?

Wait, the problem says the goal is to gather to the same point. So if the other is inactive, maybe they are already at some point, and I need to move to that point. But I don't know where they are. So that's a problem.

Alternatively, maybe the algorithm is to move in a certain pattern that covers the space, ensuring https://bramas.

period, or a way to detect the other robot's *Vait, but since we're dimensionless, maybe we can't*

Alternatively, the problem might have a standard solution. For example, in the case where two <u>hm needs to have a way to</u> robots need to gather, and they can't communicate, a common strategy is for one to stay still and the other to move, but since both are executing the same algorithm, they can't know which one to Another thought: if both robots are in the same enviror features. But since the problem doesn't specify the en be.

Interact, But the main issue is In standard robot gathering problems, some

other is a ed to ha

> y can't con e the san jorithm i

her's p

can t ires that if andle bo

> Alternatively, the algorithm could be to move in a spiral pattern, expanding outward, which would he key po eventually cover all points. If the other is active, they might also be moving in a spiral, and their paths might intersect. But this is not guaranteed.

Wait, but the problem says the goal is to gather to the same point. So maybe the algorithm needs to have a way to decide on a common point. For example, both robots use the same method to calculate a meeting point, like the midpoint between their starting positions. But if one is inactive, then the active one would go to that midpoint, but the inactive one might not be there.

In distributed systems, for two robots to gather, one common approach is for one to stay still and the other to move. But since both are executing the same algorithm, they can't know which one to be. So this leads to a deadlock.

account for the possibility that the other is inactive. So perhaps the algorithm includes a waiting

- Same algorithm
- Goal is to gather at the same point.

Wait, but if both follow the same algorithm, they might both decide to move, leading to them moving away from each other. So that's not good.

So maybe the solution is to have a randomized algorithm where each robot has a certain probability to move or stay, which increases the chance of them meeting. For example, each robot with 50% chance decides to move in a random direction, and 50% to stay still. Over time, the probability that they meet increases.

Alternatively, if they can't use randomness, maybe a deterministic algorithm that ensures meeting. But in a deterministic setting without communication, it's challenging.

Another thought: since the robots are dimensionless, maybe they can use the environment's coordinates. If there's a common coordinate system, they can agree on a meeting point. But the problem doesn't mention that.

But how would the active robot know where the other is? Unless the algorithm assumes that the other is at a certain position, which isn't given.

This is really challenging. Maybe I need to look up standard solutions for robot gathering problems. But since I can't access external information, I have to think from first principles.

ther is active, or both a th are executing the same orithm is such that the

ow like you are." So I'm could be inactive or active whether the other is active

ther is inactive, then only I

ut since the other isn't nove towards them. at I don't know. So the ation status, the gathering

the algorithm includes a w I might just keep waiting.

ive, we both move toward the other's status, how can

outed algorithms, there domized algorithm where e chance of meeting. But se the same strategy.





period, or a way to detect the other robo

- Start at the origin (0, 0). - Move 1 unit to the right, then 1 unit up, then 2 units to the left, then 2 units down, t the right, and so on, increasing the sternatively, the problem might have a standard solution. For example, in the case where two robots need to gather, and they can't communicate, a common strategy is for one to stay still and Qwen3-30b6: the other to move, but since both are executing the same algorithm, they can't know which one to

Thought Process

Okay, let's try to figure out how to approach this problem. So, me and another one. We're both executing the same algorit same point. But I don't know if the other robot is currently act do?

First, I need to understand the problem better. Since both re

To break this, the algorithm must have a way to break symmetry. This is typically done with randomness. So each robot independently decides with some probability to be the "mover" or the f one decides to move and the other stays, they meet. If both decide to move, they might that eventu move away, but with probability, they might meet eventual

So the algorithm could be: each robot, with a 50% chance, decides to move in a random direction or stay still. If they decide to move, they move in a random direction for a certain distance. If they decide to stay, they stay. Over time, the probability that one moves and the other stays increases leading to a meeting.

But since the problem says the robots are executing the same algorithm, this approach would work because both have the same probability, so eventually, one will move and the other stay, allowing

Alternatively, if the robots can sense each other, they could adjust their movement, but since the problem doesn't mention sensing, we have to assume they can't

Another possible solution is to use a "wait-for-activation" strategy. The robot waits for a certain time, and if the other is activated, they will start moving. But without knowing the other's status, this might not work

Wait, but if the other is inactive, maybe they are not moving. So the active robot could move in a certain pattern that covers the space, and if the other is inactive, the active robot will eventually reach them. But if the other is active, they would also be moving, so the paths might cross.

But how to design such a pattern? For example, moving in a spiral or a grid s this requires knowledge of the environment, which isn't specified.

Given all these possibilities, the most plausible solution seems to involve using a randomized algorithm to break symmetry, allowing the robots to eventually meet regardless of activation status. So the robot should randomly decide to move or stay, which increases the chance of meeting if the other is active, and if the other is inactive, the active robot might eventually reach them by moving in a pattern.

But I need to structure this into a coherent answer. The key points would be: use randomness to that's a pro break symmetry, so that one robot moves and the other stays, leading to a meeting. If the other is inactive, the active robot's movement will eventually cover the space and reach the inactive one.

active. but Alternative

Wait, mayb robots nee robots can But since b

But the pro So. the alg

neeting But how t

lirection elps, but owards it Alternative

are moving one goes t Wait, the p they are all

https://bramas.f Alternativel

. So perhaps the algorithm includes a w

Alternatively, the problem robots need to gather, and mment, maybe it's a 2D p

Wait, but if both follow the away from each other. So So maybe the solution is to to move or stay, which incr chance decides to move in they meet increases.

Alternatively, if they can't u But in a deterministic setti Another thought: since the coordinates. If there's a co

er's pres

es that i

Alternatively, the algorithm eventually cover all points paths might intersect. But

Wait, but the problem say have a way to decide on a calculate a meeting point, then the active one would

But how would the active ro other is at a certain position, which isn't given.

be

Wait, but if both follow the same algorithm, they might both decide to move, leading to them moving away from each other. So that's not good.

So maybe the solution is to have a randomized algorithm where each robot has a certain probability the other to move, but sinc to move or stay, which increases the chance of them meeting. For example, each robot with 50% chance decides to move in a random direction, and 50% to stay still. Over time, the probability that they meet increases

> Alternatively, if they can't use randomness, maybe a deterministic algorithm that ensures meeting. But in a deterministic setting without communication, it's challenging.

Another thought: since the robots are dimensionless, maybe they can use the environment's coordinates. If there's a common coordinate system, they can agree on a meeting point. But the problem doesn't mention that.

Alternatively, the algorithm could be to move in a spiral pattern, expanding outward, which would eventually cover all points. If the other is active, they might also be moving in a spiral, and their paths might intersect. But this is not guaranteed.

Wait, but the problem says the goal is to gather to the same point. So maybe the algorithm needs to have a way to decide on a common point. For example, both robots use the same method to problem doesn't mention th calculate a meeting point, like the midpoint between their starting positions. But if one is inactive, then the active one would go to that midpoint, but the inactive one might not be there.

> But how would the active robot know where the other is? Unless the algorithm assumes that the other is at a certain position, which isn't given.

This is really challenging. Maybe I need to look up standard solutions for robot gathering problems. But since I can't access external information, I have to think from first principles.

In distributed systems, for two robots to gather, one common approach is for one to stay still and the other to move. But since both are executing the same algorithm, they can't know which one to be. So this leads to a deadlock.

This is really challenging. Maybe I need to look up standard solutions for robot gathering problems. But since I can't access external information, I have to think from first principles.

n distributed systems, for two robots to gather, one common approach is for one to stay still and the other to move. But since both are executing the same algorithm, they can't know which one to be. So this leads to a deadlock.





- Start at the origin (0, 0). - Move 1 unit to the right, then 1 unit up, then 2 units to the left, then 2 units down, t the right, and so on, increasing the step size by 1 after every two turns. Qwen3-30b6:

Okay, let's try to figure out how to approach this problem. So her one. We're both executing the same alg same point. But I don't know if the other robot is currently a do?

First, I need to understand the problem better. Since both r

point. So maybe the algorithm needs to have a way to	mote
how?	rc
Another thought: if both robots are in the same enviro	nmŧh
features. But since the problem doesn't specify the en ar <u>soriathingi</u> onless, maybe they	vir B
r interact. But the main issue is	. w

Since both robots are executing the same algorithm and cannot communicate, a randomized approach is necessary to break symmetry and ensure eventual meeting. Here's how it works:

There but not sent the contract there drift contract of the co	
robots can cheese at direction of the other time at the other stays increases,	othoric
But since both robots are executing the same algorithm, if they are in symmetric positions, they is that the But since both robots are executing the same algorithm, if they are in symmetric positions, they is that the might move in the same way, leading to no progress. So the algorithm needs to have a way to break because both have the same probability, so eventually, one will move and the other stay, allowing symmetry, perhaps by using some randomness or a unique identifier or about the algorithm is design that en-	robots w can ⁻ sures th
But the problem states that the oborts are dimensionless they church and they don't have the international they don't have t	t the pr o handl The ke
So, the algorithmotheumber able achution of twither convert for ability induced as the store of the second method of the second	THE RE
or stay still. Overetiand, if the problem is littly in the proving a dwards a dward wither information of the problem is the problem in the problem in the problem in the problem is the problem in the problem is the problem in the problem in the problem in the problem is the problem in the p	
 But how to directions. Wait, but if the other is inactive, maybe they are not moving. So the active robot could move in a certain pattern that covers the space, and if the other is inactive, the active robot will eventually certain pattern that covers the space, and if the other is inactive, the active robot will eventually formalize them. But if the other is active, they would also be moving, so the paths might cross. helps, but maybe the inactive one is considered as a fixed point, so the active one can move towards it. But how to design such a pattern? For example, moving in a spiral or a grid search pattern. However, this requires knowledge of the environment, which isn't specified. 	
Alternatively, maybe the algorithm is to move towards a predefined point, like the origin. But if both Given all these possibilities, the most plausible solution seems to involve using a randomized are moving towards the origin, they will meet there. However if only one is active, then the active one goes to the origin but the origin active of active of a constant of the solution seems to the one goes to the origin but the origin active of a constant of the origin of the one goes to the origin but the origin active of a constant of the origin of the one goes to the origin but the origin of the ori	
Wait, the problem says tive and if the garner in the tive stars a tive to be might extend the start of the problem is a start of the problem in the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem is a start of the problem in the problem in the problem is a start of the problem in t	
in a pattern. they are already at some point, and I need to move to that point. But I don't know where they are. So that's a problemt I need to structure this into a coherent answer. The key points would be: use randomness to	
break symmetry, so that one robot moves and the other stays, leading to a meeting. If the other is Alternatively, maybe the algorithm is to move in a fertain pattern that covers the space ensuring that if the other is active, they'll eventually be in the same place. But this is vague.	
	-

Alternatively, if they can't use randomness maybe a deterministic algorithmethal ansuses precing in a spiral, and their But in a deterministic setting with out open new picetion, this is had engine to ed Another thought: since the robats are demensionless may be allowed by contraction of the syme to entry of the syme the Alternatively, the algorithm could be to move in a spiral pattern, expanding outwa eventually cover all points. If the booker of a start of the source of the start of paths might intersect. But this is not guaranteed. Wait, but the problem says the goal is to gather to the same point. So maybe the alo have a way to decide on a common point. For example, both robots us o stay still and calculate a meeting point, like the the the the set we der the relating possible on the relation one then the active one would gorts that middelift, but the mactive one middle hat be there. They can't know be. So this leads to a deadlock. But how would the active robot know where the other is? Unless the algorithm assumes that the other is at a certain position, which isn't given. This is really challenging. Maybe I need to look up standard solutions for robot gathering problems. But since I can't access external information, I have to think from first principles. In distributed systems, for two robots to gather, one common approach is for one to stay still and the other to move. But since both are executing the same algorithm, they can't know which one to be. So this leads to a deadlock.

https://bramas.

he same aligoridather they arright pointh decide to mov ernatively, the problem might have a standard solution. For example bots need to gather, and they reavine compoundents, to be wraden strategy is gorithme which we stark an at has chance decides to move in a random direction, and 50% to stay still. Over time, probability that







-> points on a 20 plane -> Look -> Compute -> more ASYNC, SSYNC & FSYNC. - no communications -> no memory -> no identifien Illo -> no common orientation.



Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns







Mobile Robots Computational Power

Landscape

RE---- Sepenato ?









Task: Rendezvous

6

Co Cn

T both robots





SSYNC



Task: Rendezvous

FSYNC Algo: move to the middle





FSYNC \bigwedge



Task: Rendezvous











Landscape

A Rendezvous

TE---- Separato 7









Landscape

SSYNC Task: move like SSYNC robots.







Flocchini, P., Santoro, N., Sudo, Y., Wada, K.: On Asynchrony, Memory, and Communication: Separations and Landscapes.

https://bramas.fr

Computational Power

Landscape

Task: move like SSYNC rebuts. is it a valid task?







Mhat is a Task ?

 $C_0 C_1 C_2 C_3 \ldots C_k C_{k+1} C_{k+2} \ldots$





$\mathbf{F} = C_0 C_1 C_2 C_3 \dots C_k C_{k+1} C_{k+1} \dots$ Predicate P(E) = Erue



What is a Task? Execution Problem





$\mathbf{F} = C_0 C_1 C_2 C_3 \dots C_k C_{k+1} C_{k+1} \dots$

Predicate P(E) = Erue

Example: Patrolling



What is a Task? Execution Problem





 $\mathbf{F} = C_0 C_1 C_2 C_3 \dots C_k C_{k+1} C_{k+1} \dots$ Predicate P true = $P(G_k) = P(G_{k+1}) = P(G_{k+1}) = \cdots$

Example: Rendez-vous

Mobile Robots Computational Power

What is a Task? Reachability Problem





Coordinate System Invariant Problem

 $\mathbf{F} = C_0 C_1 C_2 C_3 \dots C_k C_{k+1} C_{k+1} \dots$ Predicate P

Example: Pattern formation.



Mhat is a Task ?





Coordinate System Invariant Problem

Predicate DE

Example: Pattern formation.

https://bramas.fr

Mobile Robots Computational Power

Mhat is a Task ?

does not depend on the

Coordinate system.





Execution Problems Reachability Problems Coordinate System Invariant Problems









Problems

iont Problems

5





E

Τ







2 robots

(Same unit)







SYNC ⊊ SSYNC ASYNC ⊊ SSYNC ASYNC = SSYNCASYNC = SSYNC



Conclusion







Conclusion







Conclusion





https://bramas.fr

Thank you!



More details on the results



RAT Some anit

Separaton: terminate on a configuration such that $d = \frac{k}{2^{i}} \frac{3}{2} \frac{1}{k}$

ASYNC & SSYNC







R ASYNC & SSYNC









R ASYNC & SSYNC









R ASYNC & SSYNC





