Accepted Manuscript at the NETYS 2025 conference.

On Restricting Separator Problems in the OBLOT Computational Landscape

Quentin Bramas[1,2] and Sébastien Tixeuil[2]

[1] University of Strasbourg, ICUBE, CNRS, France
[2] Sorbonne Université, CNRS, LIP6, IUF

Abstract. We study the impact of restricting the class of separator problems on the look-compute-move mobile robot model hierarchy. In particular, we show that reachability specifications are sufficient to separate the semi-synchronous and the asynchronous model variants when robots agree on the unit-distance, but the two model variants are equivalent when considering terminating problems and no coordinate system agreement.

Keywords: Mobile robots · Computational hierarchy

Motivation Since the introduction of the now called *OBLOT* model by Suzuki and Yamashita [5], a number of problems have been tackled (see the reference book by Flocchini et al. [1] and references therein), aiming to characterize the minimal hypotheses (constituting the execution model) that enable problem solvability. A recent line of work explores the partial order hierarchy between the various models explored so far [3], making use of *separator problems*, that is, problems that are solvable in one model but unsolvable in another model (that differs from the first one on a single hypothesis or dimension).

A key separator problem between the semi-synchronous and the asynchronous model variants is the Monotone Line Convergence (MLCv) problem [4], that requires two robots to monotonically approach each other using the line joining them as a trajectory. As a result, specifying this problem requires one to define a predicate on the whole infinite execution of the system (a valid configuration at time t depends on every previous configuration). By contrast, most problems studied so far [1], *e.g.* pattern formation, gathering, scattering, mutual visibility, etc, can be defined by a single predicate on configurations, an execution satisfying such a specification when it is suffix-closed by configurations that satisfy the predicate. Obviously, suffix-closed specifications have weaker expressive power than full execution specifications.

Surprisingly, some well-known equivalences, such as the equivalence between semi-synchronous and asynchronous robots endowed with lights (robots have a light, visible to other robots, with a finite number of colors they can update) [2], does not hold when considering problems defined by a predicate on the entire execution, but only holds on a class of problems where a subset of the execution

satisfies a predicate (but in this class of problems, the MLCv does not separate separate semi-synchronous from asynchronous models anymore).

This dichotomy motivates the question raised in this short paper: do weaker specifications for separator problems change the computational landscape of mobile robot computing?

Model We consider the OBLOT model [1] where a set of n robots is moving on an Euclidean plane. Each robot has its own coordinate system that does not change throughout execution. The coordinate system of a robot r is associated with a *similarity* function ϕ_r (a bijection of the plane preserving angles). For a given time t, $C(t)$ denotes the positions of the robots at time t in a global fixed coordinate system (unknown to the robots).

The robots are anonymous and execute the same algorithm through Look-Compute-Move cycles. In a cycle, a robot r *Looks i.e.*, retrieve the set $\phi_r(C(t))$ of position of the other robots in its own coordinate system at time t. Then, r *Computes* a destination based on $\phi_r(C(t))$. Finally, r *Moves* towards its destination. The movements are said to be *rigid* the destination is always reached. We consider three types of schedulers: **Fully-Synchronous** ($FSYNC$) where all robots execute their cycle synchronously, **Semi-Synchronous** ($SSYNC$), where a non-empty subset of robots execute their cycle synchronously, and **Asynchronous** ($ASYNC$) where the robots execute the phases of their cycle independently. In $ASYNC$, the look phase is atomic, but the compute and move phases can take an arbitrarily long (but finite) time. This implies that a robot may look while another robot moves (or computes), hence basing its computation on a possibly outdated configuration (a robot cannot distinguish moving robots from still ones). An *execution* is an infinite sequence of configurations. We assume only *fair* executions where every robot is activated infinitely often.

We define the following set of problems encountered in the literature.

- **General problems.** The most general type of problems where the entire execution (a sequence of configurations) must satisfy a given predicate. The set of those problems is denoted $\mathcal{G}$.
- **Terminating problems.** General problems where, eventually, the configuration remains the same forever. The set of terminating problems is denoted $\mathcal{T}$.
- **Reachability problems.** Problems where the goal is to reach configurations that all satisfy a given (unique) configuration predicate. The set of reachability problems is denoted $\mathcal{R}$.
- **Similarity-invariant problems.** Reachability problems where the configuration predicate is invariant by similarity. So the predicate does not depend on a particular coordinate system, as it is the case for the pattern formation problem for instance. The set of similarity-invariant problems is denoted $\mathcal{I} \subset \mathcal{R}$.

Model Hierarchy We are interested in comparing different models. A model defines the core hypotheses that drive robots execution in a particular context. We denote by $\mathbb{M}(X_1, X_2, ...)$ the model satisfying hypotheses $X_1, X_2, \ldots$, where X_i describes an hypothesis such as a scheduler, a space where the robots evolve, a number of robots, or any other assumption.

In this short paper, we focus on models that follow the description of the $OBLOT$ model described above with only two robots and rigid movements, denoted $\mathbb{M}(2, OBLOT, rigid)$. Then, we consider two schedulers, $SSYNC$ and $ASYNC$. We may add the hypothesis, denoted $same{-}unit$, that the coordinate systems of the robots share the same unit-distance.

We say a model X is as powerful as a model Y, denoted $X \succcurlyeq Y$, if every problem that is solvable by an algorithm in model Y can also be solved by an algorithm in model X. This transitive relation defines a partial order called the **model hierarchy**. We say a model X is *more powerful* than a model Y, denoted $X \succ Y$, if it is as powerful and there exists a problem solvable in model X that is not solvable in model Y. This kind of problem is called a **separator problem**. $X \approx Y$ means $X \succcurlyeq Y \wedge Y \succcurlyeq X$.

We usually compare two models that differ only on few assumptions. For readability, if two models differ only on one hypothesis we write $\mathbb{M}(X_1, \ldots, X_k) : Y \succ Z$ instead of $\mathbb{M}(X_1, \ldots, X_k, Y) \succ \mathbb{M}(X_1, \ldots, X_k, Z)$.

In previous work, most separator problems were in class $\mathcal{G}$. The main question we want to ask is how does the problem hierarchy change when we restrict the class of allowed separator problems to (strict) subsets of $\mathcal{G}$? We define the operator $\overset{Cl}{\succ}$, resp. $\overset{Cl}{\succcurlyeq}$, as the restriction of the operator $\succ$, resp. $\succcurlyeq$, to the class of problems Cl. So, $\succ$, resp. $\succcurlyeq$ is equivalent to $\overset{\mathcal{G}}{\succ}$, resp. $\overset{\mathcal{G}}{\succcurlyeq}$.

Summary of our results. We prove the following relations:

$$\mathbb{M}(2, OBLOT, rigid, same{-}unit) : \quad SSYNC \overset{\mathcal{R}\cap\mathcal{T}}{\succ} ASYNC \tag{1}$$

$$\mathbb{M}(2, OBLOT, rigid) : \quad SSYNC \overset{\mathcal{R}}{\succ} ASYNC \tag{2}$$

$$\mathbb{M}(2, OBLOT, rigid) : \quad SSYNC \overset{\mathcal{T}\cup\mathcal{I}}{\approx} ASYNC \tag{3}$$

The separator results naturally extend to more general problem specifications. For instance, Relation (1) implies that the same relation is true considering a larger class of problems (such as $\mathcal{R}$, $\mathcal{T}$, or $\mathcal{G}$), because $SSYNC \succcurlyeq ASYNC$ is true by construction on any class, and a separator in $\mathcal{R}\cap\mathcal{T}$ is a separator in any super-set. Conversely, Relation (3) implies the same relation in more restricted classes of problems, because the existence of a separator in a subset would imply a separator in $\mathcal{T}\cup\mathcal{I}$.

Proof of (1) We consider the model $\mathbb{M}(2, OBLOT, rigid, same{-}unit)$, and we define a problem that is solvable in $SSYNC$, but not solvable in $ASYNC$.

Let $D(k, i)$ be the set of 2-robot configurations such that the two robots are at a distance $k/2^i$ for an odd integer k and an integer i. Let P be a 2-robot configuration predicate such that $P(C)$ is true if and only if the two robots are in $C \in D(k, i)$, where i is not divisible by 3. Given a team of two robots initially in a configuration in $D(k, i)$ (for any odd k and any i), the ND3 (for Non-Divisible by 3) problem consists in reaching a configuration satisfying predicate P and remaining in this configuration forever. This problem belongs to the $\mathcal{R}\cap\mathcal{T}$ class.

Let $\mathcal{A}$ be the following algorithm: Let d be the distance to the other robot, we have $d = k/2^i$ for some odd k and positive i (this is true initially and is preserved throughout the execution). If i is not divisible by three, then the activated robot

remains idle. Otherwise, the activated robot moves a distance $k/2^{i+2}$ towards the other robot.

Theorem 1. *Algorithm $\mathcal{A}$ solves the ND3 problem in* $\mathbb{M}(2, OBLOT, rigid, same-unit, SSYNC)$

Proof. If both robots are activated, then they end up at distance $k/2^{i+1}$ (which is $d - 2 \times k/2^{i+2}$), and the obtained configuration satisfies P (because $i+1$ is not divisible by 3). If only one robot is activated, then they end up at distance $(3k)/2^{i+2}$ (which is $d - k/2^{i+2}$), and the obtained configuration satisfies P (because $i+2$ is not divisible by 3).

Theorem 2. *ND3 is not solvable in* $\mathbb{M}(2, OBLOT, rigid, same-unit, ASYNC)$

Proof. Consider for the sake of contradiction that the problem can be solved with the ASYNC hypothesis by an algorithm A. Consider the configuration where the two robots are at distance $d_0 = k/2^i$ with k an odd integer, and i is divisible by 3. Consider the execution where a robot is first activated alone and let p_1 be its target at distance d_1, from the other robot. Now consider that after the activation, the robot remains in the "move" phase and is located on the path towards its destination at distance $d_1 \in (d_0, d_1)$ (or (d_1, d_0) depending on the ordering) such that $d'_1 = k'/2^{3h}$ for some integer $h \geq 1$ and odd integer k'. Then we consider that the other robot is activated. This robot sees distance d'_1 so it has to move towards a point p_2 at distance d_2. Again, we consider the execution where this robot partially performs its movement, towards a point at distance d'_2 from the target p_1 such that d'_2 is $d'_2 = k''/2^{3h'}$. When the first robot reach its destination p_1 and is activated, it sees distance d'_2 and has to move towards a point p_3 at distance d_3. We can continue this process and show that the robots are always moving and the configuration falsifies P infinitely often.

Proof of (2) Now we consider the model $\mathbb{M}(2, OBLOT, rigid)$, so robots may not agree on the distance unit. Starting from a configuration where the two robots are at distinct positions, we consider the reachability problem CD (Converge Distinct) defined by a configuration predicate P that is true if and only if the two robots are at distance less than 1 (in a global coordinate system, unknown to the robots) but not at the same point. The problem is not in $\mathcal{T}$, so it can be solved even if the robots move forever. Let $\mathcal{A}$ be the algorithm where the activated robot moves towards the other robot one third of the distance between them. Clearly, the problem is solved in $SSYNC$ by $\mathcal{A}$ because the robots keep getting closer and closer, while never reaching the same point.

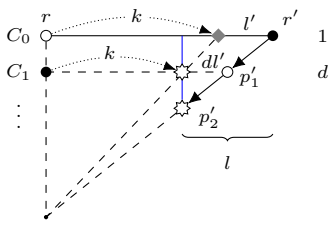

Fig. 1. Illustration accompanying the proof of Theorem 3.

Theorem 3. *CD is not solvable in* $\mathbb{M}(2, OBLOT, rigid, ASYNC)$

Proof. Consider for the sake of contradiction that the problem can be solved with the ASYNC hypothesis by an algorithm A. Consider the configuration where the two robots see themselves at distance 1. The idea of the proof, illustrated by Fig. 1, is to activate one robot (r'), then, after partially moving it (to the white circle), to activate the other robot (r) k times (for some k) to reach a configuration where the two robots are gathered. To do so, we have to carefully decide the distance d between the robot when r is activated so that its destination (the star node) is exactly the destination of r'.

First, observe that there is at least a view for which an agent moves towards the other agent. Without loss of generality, let's say that this view is when the distance between the agents is 1. So we consider that agent r, initially in C_0 has this view, is activated, and is dictated to move at distance $0 < l < 1$ to a point p'_2. Second, one can show that, if a single robot is activated infinitely often, it converges towards the other robot (otherwise we can find a round-robin execution where the distance between the robots does not tend to zero).

So, for any distance $l < 1$, after a finite number of activations, say k, robot r seeing distance 1 would end up at distance $l' < l$ (in the diamond node). Hence, if the global distance is d (in C_1), we can choose a coordinate systems for r so that it sees distance 1 and after k activations, r ends up at distance dl' from p'_1. In particular, one can show that, if we activate r when the distance between r and r' is $d = \frac{1-l}{1-l'}$, then after k activations of r (and without moving r'), r ends up at the exact destination p'_2 of r'. After finishing the movement of r', both robots occupy the same position p'_2 and cannot solve the problem anymore, a contradiction. The value of d has been found so that the top star is indeed the intersection of the lines passing through it. □

Proof of (3) Consider a problem in $\mathcal{I}$, *i.e.*, a reachability problem defined by a predicate P that is invariant by similarity. In the case of two robots, there are only four predicates that are invariant by similarity: two trivial predicates $\bot$ and $\top$, the Gathered predicate $G : \{p_1, p_2\} \mapsto p_1 = p_2$, and its negation $\bar{G} : \{p_1, p_2\} \mapsto p_1 \neq p_2$. The gathering problem and the $\bot$ are not solvable in SSYNC and the two other predicates are trivially solvable in ASYNC (just by staying idle).

Now consider a problem in $\mathcal{T}$. To prove the relation, we simply show that, either the problem is not solvable in *SSYNC*, or the problem is solvable in *ASYNC* by the idle algorithm. Assume for the sake of contradiction that the problem is solvable in *SSYNC* by a non-idle algorithm *i.e.*, an algorithm that orders at least one robot to move in at least one configuration. Then, this algorithm must order to move in all possible (non-gathered) views, otherwise, for any initial configuration we can choose the coordinate systems of the robots so that they never move, and since they must solve the problem also in this case, the idle algorithm as well, a contradiction. Since the robots are ordered to move in any non-gathered configuration, and we can make sure that the gathered configuration is never reached (by scheduling both robots if they move to the other's position, and by selecting a single robot otherwise), then the robots never terminates, a contradiction.

Open question. For each pair of model variants for which there exists a separator problem, what is the weakest possible class of the separator?

References

1. Moving and computing models: Robots. In: Flocchini, P., Prencipe, G., Santoro, N. (eds.) Distributed Computing by Mobile Entities, Current Research in Moving and Computing, Lecture Notes in Computer Science, vol. 11340, pp. 3–14. Springer (2019), https://doi.org/10.1007/978-3-030-11072-7_1
2. Das, S., Das, S., Flocchini, P., Flocchini, P., Prencipe, G., Prencipe, G., Santoro, N., Santoro, N., Yamashita, M., Yamashita, M.: The power of lights: Synchronizing asynchronous robots using visible bits. null (2012). https://doi.org/10.1109/icdcs.2012.71
3. Flocchini, P., Santoro, N., Sudo, Y., Wada, K.: On Asynchrony, Memory, and Communication: Separations and Landscapes. In: OPODIS 2023. https://doi.org/10.4230/LIPIcs.OPODIS.2023.28
4. Flocchini, P., Santoro, N., Sudo, Y., Wada, K.: On Asynchrony, Memory, and Communication: Separations and Landscapes. In: Bessani, A., Défago, X., Nakamura, J., Wada, K., Yamauchi, Y. (eds.) 27th International Conference on Principles of Distributed Systems (OPODIS 2023). Leibniz International Proceedings in Informatics (LIPIcs), vol. 286, pp. 28:1–28:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2024). https://doi.org/10.4230/LIPIcs.OPODIS.2023.28, https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.OPODIS.2023.28
5. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns. SIAM Journal on Computing **28**(4), 1347–1363 (1999)