The Power of Tokens: Rendezvous and Symmetry Detection for two Mobile Agents in a Ring

Jurek Czyzowicz[1], Stefan Dobrev[2], Evangelos Kranakis[3], and Danny Krizanc[4]

[1] Département d'informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada. Research supported in part by NSERC.
[2] School of Information Technology and Engineering, University of Ottawa, Ottawa, Canada, on leave from Slovak Academy of Sciences, Bratislava, Slovakia. Research supported in part by NSERC.
[3] School of Computer Science, Carleton University, 1125 Colonel By Drive, Ottawa, Ontario, Canada K1S 5B6. Research supported in part by NSERC and MITACS.
[4] Department of Mathematics, Wesleyan University, Middletown, Connecticut 06459, USA.

Abstract. Rendezvous with detection differs from the usual rendezvous problem in that two mobile agents not only accomplish rendezvous whenever this is possible, but can also detect the impossibility of rendezvous (e.g., due to symmetrical initial positions of the agents) in which case they are able to halt. We study the problem of rendezvous with and without detection of two anonymous mobile agents in a synchronous ring. The agents have constant memory and each of them possess one or more tokens which may be left at some nodes of the ring and noticed later. We derive sharp bounds for the case of at most two tokens per agent and also explore trade-offs between the number of tokens available to the agents and the time needed to accomplish rendezvous with detection.

1 Introduction

The mobile agent *rendezvous* problem is a search optimization problem that seeks an algorithm specifying how should anonymous mobile agents move along the vertices of a network in order to determine whether or not they can meet at some node of the network. The mobile agents are autonomous entities that move along vertices of the network acting on the information collected following a given protocol.

There are instances where the rendezvous problem is easy, e.g., if the nodes of the network have distinct identities then the mobile agents can move to a node with a specific pre-assigned identity. However, even in this case the problem becomes more difficult to solve if the agents do not have enough memory to "remember" and distinguish identities. In general, solutions of the rendezvous problem are challenging under certain conditions that delimit what the mobile agents can do and/or know about the overall status of the network. We are interested in designing rendezvous algorithms for two agents on a ring, assuming that agents have constant memory. There is a variety of interesting scenarios under consideration that may involve 1) (minimum) number of tokens used by the mobile agents, 2) knowledge of the status of the network and presence of other mobile agents in the system (e.g., number of mobile agents participating in the rendezvous problem under consideration, feasibility of rendezvous for a given starting configuration), 3) knowledge of inter-agent distances at the start, etc.

In certain instances, if the rendezvous problem is impossible to solve the mobile agents will be executing a rendezvous algorithm that may never terminate. It is therefore crucial for accomplishing rendezvous what knowledge the mobile agents have about the network configuration (e.g., size of the network, relation between start positions of the mobile agents). In this paper we distinguish between two types of rendezvous problem: *rendezvous without detection* (also simply *rendezvous*, abbreviated $\mathcal{RV}$) and *rendezvous with detection* (abbreviated $\mathcal{RD}$).

In the former case, the agents know that the rendezvous problem has a solution either for the given system configuration or regardless of the system configuration and they just want to accomplish rendezvous at a node of the ring in, say, minimum number of steps. (For example, in a ring of size n the rendezvous problem is always solvable for m mobile agents if m and n are relative primes). In the latter case, we are also interested in the decision problem which in addition to rendezvous requires a solution of the *halting problem* for rendezvous. I.e., we look for an algorithm that detects feasibility of a solution for all starting positions after a *finite number of steps* (usually dependent either on their distance or the size of the network). Thus, if rendezvous is possible then rendezvous is achieved, while if rendezvous is not possible then all agents stop and know that rendezvous is not possible.

In principle, our main concern in this study will be to accomplish rendezvous under strict constraints on the mobile agents that model their behavior. Of interest will be mobile agents that have constant memory-space (independent of the size of the ring and the number of mobile agents in the system) and thus are unable to remember either their past actions or the current conditions of the system. Thus, the conditions for the mobile agents we consider may include 1) constant memory-space and 2) a given number of tokens per mobile agent. In addition, it is assumed that the tokens are indistinguishable from each other as well as from the tokens of the other agent(s). In other words, when an agent finds a token that has been released at a node in the ring neither can it distinguish it from its own token(s) nor from the token(s) of the other mobile agents. We distinguish two cases: unidirectional ring, where agents may move in a single, say counterclockwise, direction, and bidirectional ring - where clockwise and counterclockwise moves are allowed.

1.1 Results and outline of the paper

The paper studies rendezvous with detection ($\mathcal{RD}$) and explores trade-offs between the number of tokens available to the agents and the time needed to rendezvous with detection on an n node ring. In more detail, Section 2 includes impossibility, as well as upper and lower bound results for two mobile agents with constant memory and at most two tokens each. Main results of Section 2 are summarized in Table 1. The first column depicts the number of

Conditions on		Time Required for	
Tokens #	Dirs #	$\mathcal{RD}$	$\mathcal{RV}$
1	1	∞	∞
1	2	∞	$\Theta(n^2)$
2	1	$\Theta(n^2)$	$\Theta(n^2)$
2	2	$\Theta(n^2)$	$\Theta(n^2)$

Table 1. Time bounds for two mobile agents with constant memory to detect if rendezvous is possible ($\mathcal{RD}$) and to rendezvous when input is asymmetric ($\mathcal{RV}$) on an n node synchronous uni-, bi-directional ring with one or two tokens.

tokens available per mobile agent, the second indicates the number of directions on the ring (1 means unidirectional and 2 bidirectional), while the third and fourth columns indicate the time required to solve the problem indicated. The memory required for all the algorithms depicted is $O(1)$. In Section 3 we show how the time complexity of $\mathcal{RD}$ algorithm may be improved if more than two tokens per agent are allowed. In particular, we look at the case

where each mobile agent has $t \geq 3$ tokens and memory $O(\log t)$. We give an $O(mn)$ upper bound for $\mathcal{RD}$, where m is the smallest integer such that $\binom{m-1}{t-2} \geq n-1$.

1.2 Mobile agent model

In this paper we consider an anonymous, synchronous, either unidirectional or bidirectional ring network on n nodes. The nodes of the ring are identical and do not have distinct identities. Two mobile agents are situated on the nodes of the ring. Each mobile agent owns a set of tokens that it may release at any node of the ring. The tokens are indistinguishable from each other. At any single time unit, the mobile agent occupies a node of the ring and may 1) stay at its current position, 2) move left or right from its current position, 3) detect the presence of one or more tokens at the node it is occupying, and 4) release/remove one or more tokens to/from any node it is occupying. Initially, a mobile agent may occupy any node of the network and the start node of the mobile agent is also called its *home node* or *home base*. Both agents start their algorithm at the same time. Mobile agents may communicate and exchange information with each other only when they find themselves at the same node. We say that one or more mobile agents rendezvous when either they meet at the same node or else traverse the same edge in opposite directions. They can see, remove or add tokens, and can also see each other if either they are crossing over an edge or arriving at a node at the same time. The computation is synchronous and both agents work in lock-step.

Definition 1 (Finite automaton). *More formally, the two agents holding t tokens each are starting at their home bases which are located at different nodes of a ring of unknown size n. In a bidirectional ring an agent is a finite automaton with k states represented by a* state-transition *function $\sigma : S \times T \times T \to S \times T \times \{L, R, W\}$ where $S = \{1, 2, \ldots, k\}$ and $T = \{1, 2, \ldots, 2t\}$. Moreover, whenever $\sigma(s, t_1, t_2) = (s', t_1', d)$ it must hold $t_1' \leq t_1 + t_2$. The parameters mentioned above are to be interpreted as follows:*

1. *The set S represents the agents' states, not including the number of tokens the agent holds.*
2. *$t_1 \in T$ is the number of tokens the agent holds when making the decision, and $t_2 \in T$ is the number of tokens the agents see in the current node. t_1' is the new number of tokens the agent holds after making a state transition. (Note that t_1, t_2, but also $t_1 + t_2$ are always at most $2t$, as the total number of tokens in the system is $2t$. Moreover, as the agent can pick up only the tokens it sees, the number of tokens it carries after the state transition is bounded by $t_1 + t_2$.)*
3. *L, R, W represent the "movement" decision by the agent – either moving one position* Left, Right *(according to the specified global orientation) or* Wait. *(For a unidirectional ring the moves L, R, W have to be modified to M, W meaning either* Move *or* Wait*, respectively.)*

We note that this definition of the automaton represents only a reference model and is meant to be used only as a guide. In actual proofs we will avoid strict conformance to its specific workings as it would lead to long and tedious constructions.

1.3 Related work

The rendezvous problem was first considered by the operations research community as a search game for two or more players in various network topologies (see [1], [2]). In its current algorithmic form with one token per agent the problem was first considered in [10] and more

extensively in the PhD thesis of [8]. The $\mathcal{RD}$ problem is mentioned in [7] and studied in the case of the torus in [6] where also the power of tokens for rendezvous is investigated. A related "whiteboard" model can be found in [3] whereby an agent is allowed to write messages of certain size on a white board that can be read by the other agent. So, in a sense, the token model presented here can be thought of as the "weakest possible" form of the white board model. For additional information on algorithmics for mobile agent models the reader is advised to consult [5]. General discussion of the relation of the rendezvous problem to P2P networks can be found in the edited volume [9].

2 Mobile Agents with at Most Two Tokens

In this section we consider time upper bounds for rendezvous with detection in a ring when the mobile agents have constant memory-space and at least one token each. We consider an n-node ring and two mobile agents each having a number (same for both mobile agents) of indistinguishable tokens. It goes without saying that the mobile agents cannot know the size n of the ring since they have only constant size memory, independent of n (unless n itself is a constant).

2.1 Upper bounds for rendezvous with detection

First consider the case where each mobile agent has a single token and the ring is bidirectional. In this case the asymmetric rendezvous problem is solvable, as shown in the following theorem.

Theorem 1. *In bidirectional rings, the rendezvous problem ($\mathcal{RV}$) is solvable in $O(n^2)$ time for two mobile agents having constant memory and one token each.*

Proof. Consider the following *pendulum-like* algorithm for a mobile agent:

Algorithm 1 Algorithm One-Token.

```
1: Drop your token at your home base.
2: Go right until a token is found or you meet the other agent, counting (in your state) the value of x equal
       to the distance traveled modulo three. Let this distance be x (remember it in your state).
3: if not met the other agent
4:     repeat
5:         Pick the token, reverse direction, move one step and drop the token there.
6:         Continue in current direction until a token is found, counting the value of y equal to the distance
       travelled mod3.
7:     until y ≡ (x − 1) mod 3 or met the other agent
8:     if did not meet the other agent.
9:         Stop and wait for the other agent.
10:    endif
11: endif
```

First, note that if the agents have different notion of which direction is right, they will start moving towards each other and will meet in $O(n)$ time. Hence, it is sufficient to consider the case where the agents have the same sense of direction. Let us call the agents A and B and the initial distances between them be d and $n-d$, respectively. (see Figure 1). Let $t_1, t_2, t_3, \ldots$ be the times when the agent A finds a token (i.e. t_i is the time when i-th iteration of the loop begins). Let $t'_1, t'_2, t'_3, \ldots$ be those times for the agent B. Without loss of generality

assume that $d < n - d$ and set $\delta = (n - d) - d$. Note that $d = t_1 \equiv x_A \bmod 3$ and $n - d = t'_1 \equiv x_B \bmod 3$. Observe that as long as $t_{i+1} < t'_i$, the agent B will move a token before the agent A arrives to it, and A will find the distance between the tokens has not changed. However, as $t_1 \neq t'_1$, it is easy to prove by induction that the value of $t'_i - t_i$, which is the difference between the times when both agents start their i-th iteration, increases by δ, i.e. $t'_i - t_i = i\delta$. That means that after $\lceil t_1/\delta \rceil$ iterations $t_{i+1} \geq t'_i$. If $t_{i+1} = t'_i$, the agents meet over the token. Otherwise, agent A arrives to the token before B had moved it, i.e. A traveled distance $t_1 - 1 \equiv x_A - 1 \bmod 3$. At that moment, A will stop and start waiting for B. As B has so far measured only equal distances, it will continue and eventually arrive at the place where A is waiting. Since there are t_1/δ iterations of t_1 steps each, plus final time at most t'_1 for the agent B to arrive to the meeting place, the rendezvous will happen in time $O((t_1/\delta)t_1 + t'_1)$. Since $\delta \geq 1$, $t_1 < n/2$ and $t'_1 = t_1 + \delta < n$, the resulting time is $O(n^2)$. This completes the proof of Theorem 1.

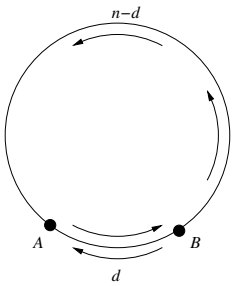

Fig. 1. Determining the rendezvous time of two mobile agents at distance d from each other.

Algorithm One-Token solves $\mathcal{RV}$ but not $\mathcal{RD}$, as it will run forever if the agents are initially in a symmetric configuration. Next we show that rendezvous with detection can be solved if we endow each agent with two tokens, even in unidirectional rings.

Theorem 2. *Rendezvous with detection ($\mathcal{RD}$) is solvable in a unidirectional ring for two mobile agents with constant memory and two tokens each, in time $O(n^2)$.*

Proof. We present an algorithm that at the cost of using two tokens per mobile agent detects the possibility of rendezvous and can eventually rendezvous when possible. Formally we have the following algorithm. Each mobile agent leaves one token at its home node and the other

Algorithm 2 Algorithm Two-Tokens.

```
1: Drop first token at your home base and second token to node located to the right.
2: repeat
3:     Travel right and move every second token you meet one position to the right.
4: until agent detects two tokens on top of each other.
5:     if two tokens are found on top of each other go around and check if other two tokens are also on top
       of each other.
6:         if yes then rendezvous is not possible else agent waits at last position.
7:         endif
8:     endif
```

token at the neighboring node located to its right. Then it travels right and moves every second token one position to the right (note that this will keep the home node tokens at their original locations). The process is repeated until the agent detects two tokens at the same node. When this happens, the agent continues traveling to check whether the other two tokens are also at a same node. If they are, then the home nodes were $n/2$ away, the whole computation was symmetric and the agents can never rendezvous. If the other pair of tokens are not at a same node, then the agent waits as the other agent will eventually come to meet it. The running time of the algorithm is as claimed. This completes the proof of Theorem 2.

Observe that the unidirectional algorithm will trivially work (with the same bound on running time) in a bidirectional ring: If the agents have the same sense of direction, they will run it as in unidirectional case, otherwise, they will start moving towards each other and meet in $O(n)$ time.

2.2 Impossibility results

In this section we investigate conditions under which the rendezvous problem ($\mathcal{RV}$) and the rendezvous with detection ($\mathcal{RD}$) are unsolvable.

As shown in Theorem 2, two tokens per agent and unidirectional ring are sufficient to solve $\mathcal{RD}$ (and hence also $\mathcal{RV}$). On the other hand, the one-token, bidirectional ring algorithm from Theorem 1 fails to solve $\mathcal{RD}$ if the initial configuration is symmetric, as it will cycle forever. Note, that $\mathcal{RD}$ is trivially solvable if agents possess $\Omega(\log n)$ memory, i.e. if the number of states exceeds the size n of the ring, even if the agents have only one token each and the ring is unidirectional. In such a case, the agent leaves the token at its home base and counts in states the distance to the token of the next agent, and checks whether this is equal to the distance to the next (his own) token. (In fact $O(\frac{\log n}{\log \log n})$ states are sufficient. See [7].) Hence, the remaining interesting cases, addressed in this section, are agents with constant memory and one token each, for either unidirectional or bidirectional rings.

Let us first introduce some tools that will be used in the impossibility and lower bound proofs. Consider an agent moving through a tokenless path of the ring. On one hand, if the agent carries no tokens, after at most $k + 1$ moves it will repeat a previously encountered state and will continue moving in the same direction until a token is encountered. On the other hand, if the agent carries a token, after at most $2k + 1$ moves two identical states will be repeated, which means that the agent has fallen into a cycle and will repeat its activity until it encounters another token. (Note that it is not necessary for the agent to carry the token the whole time – it may leave it momentarily and pick it up after a few steps and move it, but all this process is cyclically repeated, the agent is in fact carrying the token.) If no identical states with the agent carrying the token appear within the first k steps, the agent will leave the token and then it will continue moving, carrying no token, until it arrives to another token. (This idea can be extended to unidirectional rings and t tokens.) This leads us to the following definition of the *base* of an agent.

Definition 2 (Agent's Base). *A base is a contiguous segment of the ring delimited by nodes containing tokens or agents carrying a token.*

- *An* initial base *is formed when an agent leaves a token for the first time, and it consists of the single node containing this token. As the agents start in the same state, this means that initially there are two bases, at the same distance as the starting distance between the agents.*
- *When an agent holding a token leaves a base, the base expands to contain that token if and only if the agent has not entered a cyclic behavior that will make it carry the token all the way to the opposite base. Otherwise, that token ceases to be part of the base and the base shrinks to enclose its remaining tokens.*

Note that if the agents have a single token and the bases are left tokenless according to the above definition, the agents will start an infinite cycle carrying the tokens and chasing each other. If the agents have two tokens and they start carrying a token to the other base, the

bases shrink to contain a single token and the algorithm effectively resets itself (if this happens more then k times, the algorithm will cycle forever).

The next crucial parameter we need to define is important for the lower bound and impossibility proofs. It refers to the time shifts that occur in the interaction between the two agents.

Definition 3 (Time shift). *Let us denote by the increasing integer $t_1, t_2, \ldots$ the times when agent A arrives to alternating bases. More precisely, t_1 is the time when A arrives for the first time to the base of B, t_2 is the first time after t_1 that A arrives to its own base, t_3 is the first time after t_2 that A arrives to B's base). Analogously, we define the time sequence $t'_1, t'_2, \ldots$ for agent B. Let $\delta_i = |t'_i - t_i|$ and call it* time shift of round i.

Let us denote by s_A^t and s_B^t the state (including the number of tokens being carried) of agents A and B, respectively, at time t.

The following lemma forms the core of our impossibility and lower bound proofs. It states that it is possible to choose the initial distances between the agents in a large enough ring in such a way that the agents essentially arrive to the same state (but, possibly, time-shifted) for long enough time. Note that the lemma does not assume unidirectional ring.

Lemma 1. *Let the two initial distances between the agents on a ring be M and $M + xk!$ for some $M \geq k!$ and an arbitrary natural x. Then, as long as $\delta_r < M$ and the distance between the bases is at least $k+1$, it holds that $\forall i < r, \forall j \leq \min(t_{i+1} - t_i, t'_{i+1} - t'_i) : s_A^{t_i+j} = s_B^{t'_i+j}$ (and those states would be the same for any other natural number x). Moreover, at time t_i, the base to which A arrived has exactly the same configuration as the base to which agent B arrived at time t'_i.*

Proof. The proof is by induction on r. Initially, we have $t_0 = t'_0 = 0$, $s_a^0 = s_b^0$ and the initial bases have the same configuration (single node with single token each), as the agents start at the same state and at the same time. *Induction step:* Assume that $\delta_r < M$ and the distances between the bases is at least $k+1$. Moreover, assume that the base of agent A at time t_{r-1} and the base of agent B at time t'_{r-1} are in the same configuration. We will prove that the base of A at time t_r is in the same configuration as the base of B at time t'_r and that $s_A^{t_{r-1}+j} = s_B^{t'_{r-1}+j}$ for all $j \leq \min(t_r - t_{r-1}, t'_r - t'_{r-1})$.

As at time t_{r-1} agent A arrives to a base, the configuration of the base captures its state (and the same is true for agent B at time t'_{r-1}). Since the base configurations are the same (by induction hypothesis), that means that $s_A^{t_{r-1}} = s_B^{t'_{r-1}}$ and the agents arrive to the same side of the corresponding bases. The fact that the base configurations are the same also means that the agents continue behaving the same as long as the environment they see is the same. As the bases contain all tokens in the ring, the first moment when there is difference in what the agents see is when one of them arrives to the opposite base, i.e. after $\min(t_r - t_{r-1}, t'_r - t'_{r-1})$ time steps. Note that up to that moment, the configurations of both bases stayed equal (as the agents did the same modifications).

If $x = 0$, the situation is symmetric and both agents arrive to the bases at the same time and in the same state and the lemma holds, Consider now the case $x \neq 0$. Without loss of generality assume that agent A is the first one to arrive to the opposite base (at time t_r). Note that as the distance between the bases is at least $k + 1$, A must have been in a cycle (of states), moving towards the opposite base. As up to this moment B behaved the same, B

must also be in such cycle, moving towards its opposite base. This means it cannot return to the base to modify it, and the base configurations remain equal.

It remains to be shown that when agent B arrives to its opposite base at time t'_r, it will be in the same state (i.e. $s_B^{t'_r} = s_B^{t'_{r-1}+t_r-t_{r-1}} = s_A^{t_r}$). From the fact that initial distances between the bases differed by $xk!$ and the base configurations remain the same, it follows that the distances between the bases remain different by $xk!$, i.e. at time $t'_{r-1} + t_r - t_{r-1}$, B is at distance $xk!$ from its opposite base. We already know that at this moment B is moving in a cycle of states of length at most k. Let l be the forward distance traveled by B in this cycle. Obviously $l \leq k$ (it can be $l < k$ if B zig-zags). However, that means that l divides $xk!$, i.e. when B arrives to its opposite base, it will be in exactly the same state.

Lemma 1 will be used in both the unidirectional and bidirectional cases. First we consider the case of unidirectional rings.

Theorem 3. *Neither the rendezvous problem ($\mathcal{RV}$) nor rendezvous with detection ($\mathcal{RD}$) is solvable for two identical mobile agents having constant memory and one token each in a unidirectional ring.*

Proof. Consider the two mobile agents with one token each in a unidirectional ring and suppose they are represented by identical automata with k states. We will exhibit two different configurations, one symmetric and one asymmetric, that the two agents cannot distinguish. First, we consider a symmetric configuration whereby the two mobile agents start at distance $k!$ from each other in a ring of size $2k!$. (see left image in Figure 2 with $N = 0$). As long as the agents do not drop their tokens, they see the same environment (no tokens) and therefore behave (state changes and moves performed) identically. Hence, in order to rendezvous or detect that it is not possible, the agents will eventually drop their tokens, and they will be at distance $k!$ from each other at that moment.

Consider now two rings with agents starting $k!$ apart, one symmetrical of size $2k!$, the other asymmetrical of size $3k!$. Note that the conditions of Lemma 1 are satisfied, as the first ring corresponds to $M = k!$, $x = 0$, while for the second $M = k!$ and $x = 1$. Observe that because the ring is unidirectional, we get $\forall i : t_{2i} = t'_{2i}$ in both cases (even if distances between the bases are not equal, each agent alternates between traversing the short and long distance). That means that Lemma 1 holds for all r. In particular, $\forall i : s_A^{t_{2i}} = s_B^{t'_{2i}}$ and those states are the same for all x. What that means is that either the algorithm never terminates, or, if it terminates, it produces the same output for a symmetric ring of $2k!$ nodes and for the asymmetric ring of $3k!$ nodes, i.e. the algorithm is incorrect.

Next we consider the case of bidirectional rings. In view of Theorem 1 it is no longer true that $\mathcal{RV}$ is unsolvable. However, it can be shown that $\mathcal{RD}$ remains unsolvable if we limit ourselves to agents with one token and constant memory.

Theorem 4. *Rendezvous with detection ($\mathcal{RD}$) is not solvable for two identical mobile agents having constant memory and one token each in a bidirectional ring.*

Proof. The overall structure of the proof is similar to the proof of Theorem 3: we apply Lemma 1 to two configurations (one symmetric and other asymmetric) and show that either the algorithm does not terminate in the symmetric configuration, or produces wrong output in the asymmetric case. The symmetric case is a ring of size $n = 2N+2k!$ (corresponding to $M = N+k!$, $x = 0$, where N is determined later), the asymmetric case uses the same M but $x = 1$.

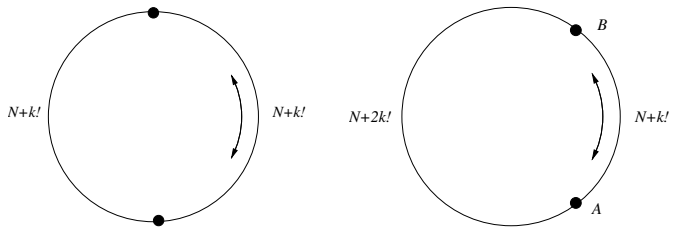

Fig. 2. Unsolvability of rendezvous with detection in a unidirectional ring. Left image represents a ring of size $2N+2k!$ and the right image of size $2N+3k!$.

Assume that the algorithm correctly decides in the symmetric ring, after r rounds (r, cf. Lemma 1, corresponds to how many times an agent switched bases). Since the agents have k states, we get that r must be at most k (otherwise the algorithm would cycle forever). We will define N in such a way that Lemma 1 applies for at least $k+1$ rounds (for the faster agent) in the asymmetric case. That would ensure that in the asymmetric execution the same deciding state appears and the algorithm decides incorrectly.

How much can δ_i grow in each round? The only difference the agents experience is that the distances between the bases differ by $k!$. In one round, each agent crosses this distance exactly once. Let c represent the maximal time it takes an agent to travel one step (it can be more than one, as the agent can zig-zag, but it is a constant as the agent has k states). Then, each round the time shift can grow by at most $ck!$ and after i rounds the time shift is at most $ick!$. Choosing $N > ic(k+1)!$ ensures that Lemma 1 applies for at least $k+1$ rounds.

2.3 Lower bounds for rendezvous

In Section 2.1 we have shown $O(n^2)$ upper bounds for settings not excluded by the impossibility results from Section 2.2. An obvious question to ask is whether these upper bounds are tight. In this section we provide an affirmative answer by proving an $\Omega(n^2)$ lower bound for $\mathcal{RV}$ in bidirectional rings for agents with two tokens. This implies the same lower bound also for $\mathcal{RD}$, as well as for unidirectional rings and single-token agents. The main result of this section is the following theorem.

Theorem 5. *The rendezvous problem ($\mathcal{RV}$) for two mobile agents having constant memory and two tokens each require $\Omega(n^2)$ time in a bidirectional ring of size n.*

Proof. Consider a ring of size $n = 2N + k!$ with the agents starting at distance N from each other. Again, Lemma 1 applies while $\delta_r < N$ and the distance between the bases is at least $k+1$. We show that (1) if $\delta_r < N/3$ holds until the bases reach size $N/3$, then it takes $\Omega(N^2)$ time to increase the bases to size $N/3$, and (2) if the base sizes are at most $N/3$, then it takes $\Omega(N^2)$ time to increase δ_r to $N/3$. When combined with a choice of $N \geq k!$ this yields the desired $\Omega(n^2)$ lower bound.

Assume first that the bases reach size $N/3$ before δ_r exceeds $N/3$. Note that at the moment a base has 2 tokens for the first time, the size of the base is at most k (otherwise an agent is carrying the second token in a cycle and by definition it is not a part of the base). Moreover, an agent can increase the size of the base by at most k before it enters a cyclic behavior – which will either take it to the other base, or to the other end of the current base.

Let x_j be the size of the base at the moment an agent is leaving the base's endpoint in a cycle for the j-th time (called *quasi-round* j). Let p_j and p'_j, respectively, be the times this happens for the two bases. We get that $x_{j+1} \leq x_j + k$ and $p_{j+1} \geq p_j + x_j$: The base grows

either when an agent traverses it from one end to another (obviously taking x_j time), or when an agent moves from one base to another, yielding $p_{j+1} \geq p'_j + (N - x_j)$. As the time difference is at most $N/3$ and $x_j \leq N/3$, we get $p_{j+1} \geq p'_j + (N - x_j) \geq p'_j + 2N/3 \geq p_j + N/3 \geq p_j + x_j$. As $x_{j+1} - x_j \leq k$ there must be at least $N/3k$ quasi-rounds. Let f be the final quasi-round, i.e. $x_f \geq N/3$. Summing over all rounds we get $p_f = \sum_{j=1}^{f} x_j = \sum_{j=0}^{f-1} x_{f-j} \geq \sum_{j=0}^{f-1} (N/3 - jk) \in \Omega(N^2)$, as $f \geq N/3k$ and k is a constant.

Consider now the case that δ_r exceeds $N/3$ before the bases reach size $N/3$. Using the same argument as in the proof of Theorem 4, we get that $\delta_i \leq ick!$, i.e. $r \geq N/3ck!$. As the base sizes are at most $N/3$, each round takes at least $2N/3$ time steps (each round involves traversing from one base to another, by definition of a round). Summing up over all r rounds we get the lower bound $N^2(2/9)ck!$, which is $\Omega(n^2)$, since c and k are constants.

3 Mobile Agents with More than Two Tokens

Now that we have the complete picture for rendezvous with detection when each mobile agent has at most two tokens we look for the more general case whereby each mobile agent has more than two tokens.

3.1 Upper bounds for rendezvous with detection

The first theorem provides a trade-off between number t of tokens being used and time required for $\mathcal{RD}$. We consider the case of at least three tokens per mobile agent, i.e., $t \geq 3$.

Theorem 6. *Consider a synchronous, bidirectional ring with n nodes and two mobile agents located at two of its nodes. Rendezvous with detection ($\mathcal{RD}$) is solvable for two mobile agents having $t \geq 3$ tokens and $O(\log t)$ bits of memory each in time $O(mn)$, where m is the smallest integer such that $\binom{m-1}{t-2} \geq n - 1$.*

Proof. A basic idea of the algorithm is to implement a counter C_t, that can count up to n. The counter will be represented by a segment of nodes of the ring containing up to t tokens at its nodes, delimited by two tokens at a maximum distance m, say, from each other. The values that this counter takes are held within this segment of nodes of the ring; one of the two tokens delimiting it is located at the home base of the agent while the other is the last token that the mobile agent released at a node of the ring at distance m from its home base.

Assuming that such counter exists, we can proceed just like in the proof of Theorem 2. The basic idea for $\mathcal{RD}$ is to have an agent go from its home base to the home base of the other agent, while incrementing its counter by one once in each round. After the counter reaches its maximum, the agent continues to go from its home base to the base of the other agent but now decrementing its value by one once in each round. Notice that the counter can be incremented/decremented in time $O(m)$ per round. As with the algorithm of Theorem 2, when the counter reaches 0 before encountering its own home base, the mobile agent goes to the first base, and if the counter of the other mobile agent reaches exactly 0 at the second base as well, then the situation is symmetric and rendezvous is impossible. Otherwise this agent waits at the second base until the other agent comes there and the rendezvous is accomplished. Therefore the algorithm whose idea has just been described not only reaches rendezvous, whenever possible, but also detects when it is not possible within the same time bound. The running time of the algorithm presented will be $O(mn)$ since each round takes n steps which is the size of the ring.

Clearly, the running time of the algorithm is directly proportional to how compact the counter C_t can be, as the cost of moving is proportional to its size m, which is the distance between the two tokens delimiting the counter C_t. Therefore for the given number t of tokens it remains to determine m so that the mobile agent can implement a counter that can hold a maximum value n. By assumption, each mobile agent has t tokens. One token is being used to mark the mobile agent's home base thus leaving $t-1$ tokens that can be used to implement the counter. The technical part is how to implement the counter, with the remaining $t-1$ tokens. The counter will be delimited by two tokens, located in nodes A, B, at distance m apart. A token is located at the home base A, say, of an agent. This leaves $t-1$ tokens for marking positions at nodes of the network. Another token located at B increments the range of the counter. Since the agent can count internally to $t-1$ (since it has $t-1$ tokens), all possible combinations of $t-3$ tokens between two fencing tokens at distance k can be tried, and afterward increment k and repeat until the home base of the other mobile agent is reached, where $k \leq m$.

It remains to investigate what the size of the counter should be so as to guarantee that it is able to count up to n. For given k, there are $\binom{k-2}{t-3}$ possibilities (assuming no two tokens can can be left at the same node, but appropriate combination numbers can be derived for that as well). Summing up over all $k \leq m$ until the home base of the other agent is reached results in at most $\sum_{k=2}^{m} \binom{k-2}{t-3} = \binom{m-1}{t-2}$ possibilities (see [4][page 56]). Since the position of the home base of the other agent is at most $n-1$ the counter C_t needs to count up to $n-1$. Therefore the value of m will never need to exceed the smallest m such that $\binom{m-1}{t-2} \geq n-1$. Further, notice that the two agents are required to have $O(\log t)$ bits of memory so that they can count internally up to t and thus distinguish the two delimiters of the counter C_t. This completes the proof of Theorem 6.

As a corollary of Theorem 6 we obtain the following result.

Corollary 1. *Rendezvous with detection ($\mathcal{RD}$) is solvable for two mobile agents having $t > 2$ tokens and memory $O(\log t)$ each, in time $O(n^{\frac{t-1}{t-2}} t)$ in a bidirectional ring. Moreover, if $t = \log n$ then the algorithm works in time $O(n \log n)$.*

3.2 Lower bounds for rendezvous in unidirectional rings

Very little is known concerning lower bounds for agents with more than two tokens. However, for the case of unidirectional rings we can improve on the result of Theorem 5 thus relating the rendezvous time with the number of tokens available to each agent.

Theorem 7. *The rendezvous problem ($\mathcal{RV}$) for two mobile agents having constant memory and t tokens each requires $\Omega(n^2/t)$ time in an unidirectional ring of size n. Moreover, there is an algorithm achieving this bound.*

Proof. This is similar to the proof of Theorem 5. As before, we take a ring of size $n = 2N + k!$, with the agents starting at distance N. Because the ring is unidirectional, $\delta_{2i} = 0$ and $\delta_{2i+1} = k!$, as in two consecutive rounds each agent traverses the entire ring. Moreover, the quasi-rounds correspond to the rounds from Lemma 1, as an agent cannot reverse direction and traverse and increase the size of the base it have just crossed. An agent having t tokens can increase the size of the base by at most $(t-1)k$: one token should remain to mark the beginning of the base, so an agent carries at any moment at most $t-1$ tokens. In the $(t-1)k+1$ steps

after the agent left the right endpoint of the former base, at least two agent configurations (state, number of tokens held) repeat. By the definition of a base, the new base stops growing when the agent starts cycling, i.e. $x_{j+1} \leq x_j + (t-1)k$. That means that Lemma 1 holds for at least $N/(t-1)k$ rounds. As two consecutive rounds last together n time steps, we get that the algorithm cannot terminate before time $nN/(2(t-1)k \in O(n^2/t)$.

The matching upper bound is simple. Each agent goes around and every round it increases the base by t: skip first token of the base, then walk until the second token is found, pick it up and keep picking up until you have $t-1$ tokens in hand (if you cannot count up to t, just have the tokens next to each other, the first empty space means end-of-base), and then just lay them down one after another. While leaving the tokens of your base, verify if you fall onto the other agent's base.

4 Conclusion and open problems

In this paper we studied the rendezvous problem $\mathcal{RV}$ and rendezvous with detection $\mathcal{RD}$ for two variants of a synchronous, anonymous ring: unidirectional and bidirectional. We considered two mobile agents with constant memory (i.e. unrelated to the size of the ring) and derived several sharp upper and lower bounds when the agents have at most two tokens each and also studied the case of multiple tokens. This enabled us to characterize the power of tokens for rendezvous in such a setting. Several challenging problems remain. Some of them concern closing gaps remaining in the trade-offs derived in this paper. Generally, we are lacking general non-trivial lower bounds for $t \geq 3$ tokens. E.g., can we derive sharp upper and lower bounds for t tokens? Another problem is related to the case $t = 3$: are three tokens really more powerful than two tokens (see Theorem 6)? It would also be interesting to look at rendezvous with detection for more than two mobile agents, and also consider the case where no synchrony is assumed.

References

1. S. Alpern. Rendezvous search: A personal perspective. *Operations Research*, 50(5):772–795, 2002.
2. S. Alpern and S. Gal. *The Theory of Search Games and Rendezvous*. Kluwer Academic Publishers, Norwell, Massachusetts, 2003.
3. S. Dobrev, P. Flocchini, G. Prencipe, and N. Santoro. Multiple agents rendezvous in a ring in spite of a black hole. In *Symposium on Principles of Distributed Systems (OPODIS '03)*, pages 34–46, 2004. Springer Verlag LNCS.
4. D. Knuth. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. Addison-Wesley, Reading, MA., 1997. 3rd edition.
5. E. Kranakis and D. Krizanc. An algorithmic theory of mobile agents. In *Proceedings of TGC 2006, 2nd Symposium on Trustworthy Global Computing (Lucca, Italy, November 7 - 9)*, volume 4661. Springer Verlag LNCS, 2007.
6. E. Kranakis, D. Krizanc, and E. Markou. Mobile agent rendezvous in a synchronous torus. In *Proceedings of LATIN 2006, March 20-24, Valdivia, Chile*, volume 3887. Springer Verlag LNCS, 2006.
7. E. Kranakis, D. Krizanc, N. Santoro, and C. Sawchuk. Mobile agent rendezvous search problem in the ring. In *International Conference on Distributed Computing Systems (ICDCS)*, pages 592–599, 2003.
8. C. Sawchuk. *Mobile Agent Rendezvous in the Ring*. PhD thesis, Carleton University, School of Computer Science, Ottawa, Canada, 2004.
9. G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Wiley, 2002.
10. X. Yu and M. Yung. Agent rendezvous: A dynamic symmetry-breaking problem. In *Proceedings of ICALP '96*, pages 610–621. Springer Verlag LNCS, 1996.