Finite-State Robots in the Land of Rationalia

Arnold L. Rosenberg

Northeastern University, Boston, MA 02115, USA,
rsnbrg@ccs.neu.edu

Abstract. Advancing technologies have enabled simple mobile robots that collaborate to perform complex tasks. Understanding how to achieve such collaboration with *simpler* robots leverages these advances, potentially allowing more robots for a given cost and/or decreasing the cost of deploying a fixed number of robots. This paper is a step toward understanding the algorithmic strengths and weaknesses of robots that are identical mobile *finite-state machines* (*FSM*s)—FSMs being the avatar of "simple" digital computers. We study the ability of (teams of) FSMs to *identify* and *search within* varied-size quadrants of square ($n \times n$) meshes of tiles—such meshes being the avatars of tesselated geographically constrained environments. Each team must accomplish its assigned tasks *scalably*—i.e., in arbitrarily large meshes (equivalently, for arbitrarily large values of n). Each subdivision of a mesh into quadrants is specified via a pair of fractions $\langle \varphi, \psi \rangle$, where $0 < \varphi, \psi < 1$, chosen from a *fixed, finite* repertoire of such pairs. The quadrants specified by the pair $\langle \varphi, \psi \rangle$ are delimited by a horizontal line and a vertical line that cross at *anchor* mesh-tile $v^{(\varphi,\psi)} = \langle \lfloor \varphi(n-1) \rfloor, \lfloor \psi(n-1) \rfloor \rangle$. The current results:

- *A single FSM cannot identify tile $v^{(\varphi,\psi)}$ in meshes of arbitrary sizes, even for a single pair $\langle \varphi, \psi \rangle$*—except *when $v^{(\varphi,\psi)}$ resides on a mesh-edge.*
- *A pair of identical FSMs can identify tiles $v^{(\varphi_i,\psi_i)}$ in meshes of arbitrary sizes, for arbitrary fixed finite sets of k pairs $\{\langle \varphi_i, \psi_i \rangle\}_{i=1}^{k}$. The pair can sweep each of the resulting quadrants in turn.*
- *Single FSMs can always* verify *(for all pairs and meshes) that all of the tiles of each quadrant are labeled in a way that is unique to that quadrant. This process parallelizes linearly for teams of FSMs.*

Keywords: Finite-state mobile robots, path planning/exploration

1 A Motivating Story

Managing Agriculture in Rationalia. The state of Rationalia controls its agrarian economy very tightly. Years ago, the state partitioned all arable land into 1×1 *unit-plots* whose (common) physical size is dictated by the demands of the agricultural endeavor: the need to cultivate, plant, and harvest each unit-plot. Each year, after reviewing all farmers' performances (yields, cost efficiencies, etc.), the state aggregates the unit-plots into square plots and allocates to each farmer Φ_i a plot of dimensions $n_i \times n_i$ (measured in unit-plots), where n_i is determined based on Φ_i's past performance. (The unit-plots of each $n_i \times n_i$

plot are indexed from $\langle 0,0\rangle$ in the northwest corner to $\langle n_i-1, n_i-1\rangle$ in the southeast.) Each farmer cultivates crops of the same 4 types, which we label A, B, C, D. (Clerical extensions will handle k crop-types.) As with plot sizes, the government uses past performance to determine how much of each crop-type each farmer should cultivate. Formally, each Φ_i is assigned a fixed pair $\langle \varphi_i, \psi_i\rangle$ of *rational numbers* (what else, given the country's name?), each strictly between 0 and 1. Φ_i's $n_i \times n_i$ plot is then partitioned into quadrants determined by the pair $\langle \varphi_i, \psi_i\rangle$. Each farmer's quadrant NW is devoted to crop-type A, quadrant NE to crop-type B, quadrant SW to crop-type C, and quadrant SE to crop-type D. In detail, each Φ_i's quadrants are specified by passing through her $n_i \times n_i$ plot a horizontal line and a vertical line that cross at the *anchor unit-plot* $v_i = \langle \lfloor \varphi_i(n_i-1)\rfloor, \lfloor \psi_i(n_i-1)\rfloor\rangle$, leading to the pattern depicted in Fig. 1. Anchor unit-plot v_i determines where Φ_i's quadrants meet and the crop-types change.

	$\lfloor \varphi_i(n_i-1)\rfloor$ columns	$n_i - \lfloor \varphi_i(n_i-1)\rfloor$ columns
$\lfloor \psi_i(n_i-1)\rfloor$ rows	all of type A	all of type B
$n_i - \lfloor \psi_i(n_i-1)\rfloor$ rows	all of type C	all of type D

Fig. 1. The arrangement of Φ_i's crop-types; lengths count unit-plots.

To implement the described system, the government must efficiently partition each farmer's plot into quadrants and sow each quadrant's unit-plots with crops of the appropriate type, achieving the arrangement of Fig. 1. We formalize this organization problem via: the *Anchor-Identification (A-I) Problem,* which requires the organizing agent(s) to *identify* each farmer's anchor unit-plot; the *Plot-Sweep (P-S) Problem,* which requires the agent(s) to sweep each of the resulting quadrants, in turn, sowing the authorized type of crop in each. The government faces an additional challenge. Regrettably, some farmers cheat in order to increase their profits, specifically by changing their allocation parameters $\langle \varphi_i, \psi_i\rangle$ in response to the relative profitability of the crop-types. The government must *monitor* each farmer's compliance with her assigned allocation parameters. This is the *Compliance-Checking (C-C) Problem.*

Complicating the preceding tasks is the government's extreme reluctance to expend money. Therefore, it convened an *elite task force* to determine:
1. How little intelligence do robots need to solve our three Problems?
2. Given the preceding bounds, how few robots suffice to solve the Problems?
The government's operating assumptions are:
– Employing robots would be cheaper than employing humans.
– Less-"intelligent" robots are less expensive to deploy than more capable ones.
In this paper, we play the role of the *elite task force.* We craft a formal setting for the preceding story and study whether robots that have the capabilities (or, "intelligence") of *finite-state machines* (*FSM*s) can accomplish the following formalized versions of the three Problems for arbitrary anchor unit-plots v.
1. *The A-I Problem:* FSM(s) proceed from their initial unit-plots to v.

2. *The P-S Problem:* FSM(s) sweep each quadrant specified by v and label each encountered unit-plot u with its assigned crop-type.
3. *The C-C Problem:* FSM(s) sweep the plot and check that each encountered unit-plot u has the authorized label.
We view FSMs as the lowest level of "intelligence" that might be able to solve the preceding Problems. Informally, we show that:

1. *A single FSM cannot solve the A-I Problem in arbitrary plots—even for a single pair of parameters $\langle \varphi, \psi \rangle$. Not obviously, a single FSM* can *solve the A-I Problem when the anchor unit-plot resides on an edge of the plot.*
2. *A team of ≥ 2 identical FSMs can solve the A-I Problem in arbitrary plots, for arbitrary fixed pairs of parameters. Having discovered an anchor unit-plot, the team can solve the P-S Problem for the resulting quadrants.*
3. *A single FSM can solve arbitrary instances of the C-C Problem; $k > 1$ identical FSMs can accomplish this k times faster than a single FSM (to within rounding).*

2 Technical Background and Related Work

2.1 Technical background. Our model of *FSM-robot* (*FSM*) augments the capabilities of standard finite-state machines (see, e.g., [16]) with the ability to navigate square *meshes* (our story's "plots") of *tiles* (our story's "unit-plots").
Meshes and tiles. Every edge of every tile v is labeled to indicate which of v's potentially four neighbors actually exist. (Labels on tile edges enable FSMs to avoid "falling off" $\mathcal{M}_n$ by moving to a nonexistent tile.) $\mathcal{M}_n$ admits partitions into *quadrants* (labeled NW, NE, SE, SW in clockwise order) that are determined by crossing lines perpendicular to its edges; each partition is determined by an *anchor tile* at which the defining horizontal and vertical line cross.
A single FSM on $\mathcal{M}_n$. At any moment, an FSM $\mathcal{F}$ occupies a single tile of $\mathcal{M}_n$, coresiding with the crop in that tile *but with no other FSM.* At each step, $\mathcal{F}$ can move to any of the (≤ 4) neighbors of its current tile in the primary compass directions: (N)orth, (E)ast, (W)est, (S)outh. (One easily augments $\mathcal{F}$'s move repertoire with any *fixed finite* set of atomic moves.) As $\mathcal{F}$ plans its next move, it *must* consider the label of its current tile—to avoid "falling off" $\mathcal{M}_n$.
Multiple FSMs on $\mathcal{M}_n$. All FSMs operate synchronously, hence, can follow trajectories *in lockstep.* This ability is no less realistic than are human synchronous-start endeavors. FSMs on neighboring tiles can exchange (simple) messages, e.g., "I AM HERE." This enables one FSM to act as an "usher" for others; cf. Sec. 4.
FSMs' moves are tightly orchestrated: an FSM attempts to move in direction:

N only at steps $t \equiv 0 \bmod 4$; E only at steps $t \equiv 1 \bmod 4$;
S only at steps $t \equiv 2 \bmod 4$; W only at steps $t \equiv 3 \bmod 4$

(Larger repertoires of atomic moves require larger moduli.) Thereby, *FSMs need never collide!* If several FSMs want to enter a tile from (perforce distinct) neighboring tiles, then one will have permission to enter before the others.

2.2 Algorithmic standards.
• *Algorithms are scalable.* They work on arbitrary-size meshes; FSMs can learn

only "finite-state" properties of $\mathcal{M}_n$'s size measures (n, n^2)—e.g., parity.
• *All FSMs are identical:* (a) None has a "name" that renders it unique. (b) All execute the same *finite-state* program; cf. [16, 19].
These standards are often violated in implementations of "ant-like" robots (cf. [8, 11, 18]), where practical simplicity overshadows algorithmic simplicity.

2.3 Related work. Our study combines ideas from complementary bodies of literature that span several decades. The literature on automata theory and its applications contains studies such as [3–5, 7, 14] that focus on the (in)ability of FSMs to explore graphs with goals such as finding "entrance"-to-"exit" paths or exhaustively visiting all nodes or all edges. Other studies, e.g., [10], focus on algorithms that enable FSMs that populate the tiles of (multidimensional) meshes—the *cellular automaton* model [9]—to synchronize. The robotics literature contains numerous studies—e.g., [1, 2, 8, 18]—that explore ants as a metaphor for simple robots that collaborate to accomplish complex tasks. Cellular automata appear in many application- and implementation-oriented robotic applications of automata-theory [11, 13, 15, 18]. The current study melds the automata-theoretic and robotic points of view by studying FSMs that traverse square meshes, with goals more closely motivated by robotics than automata theory. Our closest precursor is [17], which requires each FSM in a mesh to *park*, i.e., go to its closest corner and organize with other FSMs there into a maximally compact formation.

3 The Anchor-Identification Problem

The *Anchor-Identification (A-I) Problem* for $\mathcal{M}_n$ is formalized as follows.

Input: A pair of rationals $\langle \varphi, \psi \rangle$, where $0 < \varphi, \psi < 1$
Task: All FSMs on $\mathcal{M}_n$ proceed from their initial tiles to anchor tile $v^{(\varphi,\psi)}$. One FSM ends on $v^{(\varphi,\psi)}$; all others cluster around it.

3.1 The general A-I Problem for one FSM. *A single FSM cannot solve the A-I Problem on arbitrarily large meshes, for any input pair* $\langle \varphi, \psi \rangle$. The proof exploits a result from [17] that exposes the inability of single FSMs to navigate the *interiors* of large meshes, i.e., submeshes that are bounded away from the edges.

3.2 The Edge-Constrained A-I Problem for one FSM. Not obviously, a single FSM *can* solve the variant of the A-I Problem wherein the sought anchor-tile resides on an edge of $\mathcal{M}_n$. Formally, this Problem for FSM $\mathcal{F}$ is:

Input: A rational φ with $0 < \varphi < 1$
Task: $\mathcal{F}$ proceeds from its initial tile to:
$$\begin{cases} \text{bottom version:} & \langle (n-1), \lfloor \varphi(n-1) \rfloor \rangle \\ \text{top version:} & \langle 0, \lfloor \varphi(n-1) \rfloor \rangle \\ \text{left version:} & \langle \lfloor \varphi(n-1) \rfloor, 0 \rangle \\ \text{right version:} & \langle \lfloor \varphi(n-1) \rfloor, (n-1) \rangle \end{cases}$$

Theorem 1. *For any fixed rational* $0 < \varphi < 1$*: A single FSM* $\mathcal{F}^{(\varphi)}$ *whose size depends only on* φ *can solve the Edge-Constrained A-I Problem with input* φ *on any mesh* $\mathcal{M}_n$*, within* $O(n)$ *steps.*

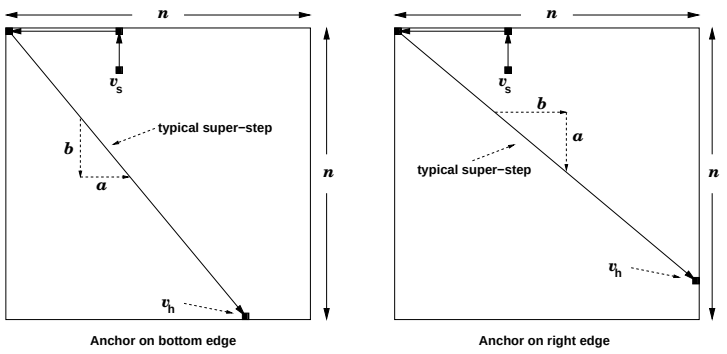

Fig. 2. $\mathcal{F}(\varphi)$ proceeds from tile $v_{\rm s}$ to the anchor tile $v_{\rm h}$ to solve: (left) the bottom-edge-Constrained A-I Problem and (right) the right-edge version, both with input $\varphi = a/b$.

Sketch. Let $\varphi = a/b$ for integers $b > a > 0$. $\mathcal{F}^{(\varphi)}$ moves from $v_{\rm s}$ to $v_{\rm h}$: it goes to $\langle 0, 0\rangle$ and continues thence to $v_{\rm h}$ via a diagonal walk of *super-steps*. The *bottom-edge* walk has slope $-b/a$; the *right-edge* walk has slope $-a/b$ (Fig. 2). □

3.3 The A-I Problem for teams of (≥ 2) identical FSMs.

Theorem 2. *Let $\Psi = \{\langle\varphi_i, \psi_i\rangle\}_{i=1}^{k}$ be any fixed set of rational pairs, where $0 < \varphi_i, \psi_j < 1$ for all i, j. One can design an FSM $\mathcal{F}^{(\Psi)}$ such that a team of two or more copies of $\mathcal{F}^{(\Psi)}$ can solve the A-I Problem in every mesh $\mathcal{M}_n$, for every pair $\langle\varphi, \psi\rangle \in \Psi$, within $O(n)$ synchronous steps.*

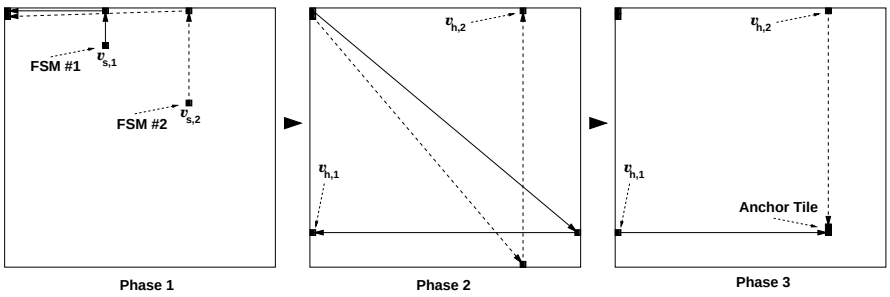

Fig. 3. The 3-phase coordinated trajectories for FSMs $\mathcal{F}_1$ and $\mathcal{F}_2$ to solve the A-I Problem with inputs $\langle\varphi, \psi\rangle$. Solid lines show $\mathcal{F}_1$'s trajectory; dashed lines show $\mathcal{F}_2$'s.

Sketch. We use Theorem 1 to design identical FSMs $\mathcal{F}_1$ and $\mathcal{F}_2$ that solve the A-I Problem for a rational pair $\langle\varphi, \psi\rangle$. See Fig. 3. $\mathcal{F}_1$ and $\mathcal{F}_2$ meet at tile $\langle 0, 0\rangle$, then execute the algorithm of Theorem 1 to go to the projections of anchor $v^{(\varphi,\psi)}$: $\mathcal{F}_1$ goes to the left-edge anchor $v_{\rm h,1}$; $\mathcal{F}_2$ goes (*in lockstep*) to the top-edge anchor

$v_{\mathrm{h},2}$. When $\mathcal{F}_1$ reaches $v_{\mathrm{h},1}$ (resp., $\mathcal{F}_2$ reaches $v_{\mathrm{h},2}$), it starts to walk eastward (resp., delays one step, then starts to walk southward). The FSMs halt when they meet: $\mathcal{F}_1$ is then on tile $v^{(\varphi,\psi)}$; $\mathcal{F}_2$ is on $v^{(\varphi,\psi)}$'s northward neighbor. □

4 The Plot-Sweep Problem

A pair of FSMs sweep through each quadrant specified by anchor tile v, in turn.

Theorem 3. *For any rational pair $\langle\varphi,\psi\rangle$, where $0 < \varphi, \psi < 1$, one can design an FSM $\mathcal{F}^{(\varphi,\psi)}$ such that team of two copies of $\mathcal{F}^{(\varphi,\psi)}$ can solve the P-S Problem in every mesh $\mathcal{M}_n$.*

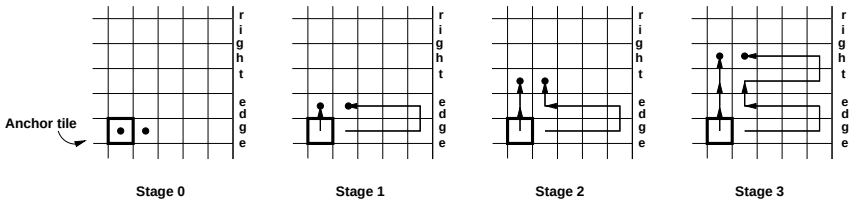

Fig. 4. Starting a sweep of $\mathcal{M}_n$'s NE quadrant. The left FSM "ushers" the right one.

Sketch. Focus on a sweep of quadrant NE. Once $\mathcal{F}_1$ and $\mathcal{F}_2$ identify anchor tile $v^{(\varphi,\psi)}$, $\mathcal{F}_1$ moves to $v^{(\varphi,\psi)}$ and $\mathcal{F}_2$ to $v^{(\varphi,\psi)}$'s eastward neighbor (stage 0 of Fig. 4). Thence, $\mathcal{F}_1$ climbs the column that extends northward from $v^{(\varphi,\psi)}$ and acts as an "usher" while $\mathcal{F}_2$ threads the rest of the quadrant; see Fig. 4. □

5 Compliance-Checking by a Single FSM

Our final task is the *Compliance-Checking (C-C) Problem* for $\mathcal{M}_n$:

Input: A pair or rationals $\langle\varphi,\psi\rangle$, where $0 < \varphi, \psi < 1$
Task: The FSM(s) on $\mathcal{M}_n$ perform a sweep from tile $\langle 0,0\rangle$, to check that the tile labels have the format illustrated in Fig. 1.

Theorem 4. *A team of $k \geq 1$ identical FSMs can solve the C-C Problem for any pair of rationals $\langle\varphi,\psi\rangle$, in any mesh $\mathcal{M}_n$, within $\frac{1}{k}n^2 + O(n)$ steps.*

Sketch. An FSM $\mathcal{F}^{(\langle\varphi,\psi\rangle)}$ can check that boundaries are as in Fig. 1 in two phases. Using a *row-sweep*, $\mathcal{F}^{(\langle\varphi,\psi\rangle)}$ easily checks that each row's crop-labels belong to $A^\star B^\star$ or to $C^\star D^\star$; see Fig. 5(left) for the case $k = 1$. (Larger teams use the Edge-Constrained A-I algorithm to partition the columns evenly among them.) In the *boundary* phase, $\mathcal{F}^{(\langle\varphi,\psi\rangle)}$ finds the *edge-constrained projections* of anchor tile $v^{(\varphi,\psi)}$, viz., tiles $v_{\mathrm{horiz}} = \langle\lfloor\varphi(n-1)\rfloor, n-1\rangle$ and $v_{\mathrm{vert}} = \langle n-1, \lfloor\psi(n-1)\rfloor\rangle$. Sawtooth trajectories northward from v_{horiz} and westward from v_{vert} enable $\mathcal{F}^{(\varphi,\psi)}$ to verify all boundaries (Fig. 5(middle, right)). □

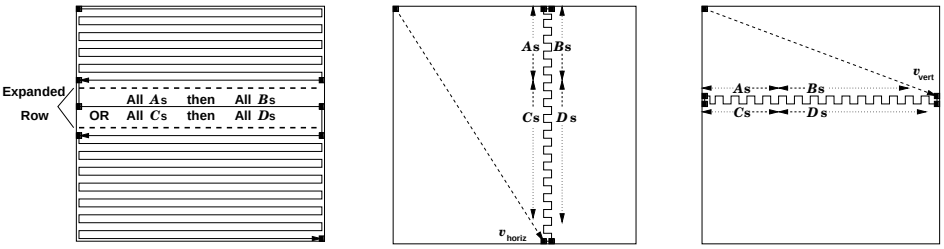

Fig. 5. (left) $\mathcal{F}^{(\varphi,\psi)}$ verifies the form of each row's labels: (left to right) all As followed by all Bs *OR* all Cs followed by all Ds. (middle, right) $\mathcal{F}^{(\varphi,\psi)}$ verifies that labeled quadrants have the correct horizontal and vertical endpoints.

6 Conclusions

6.1 Retrospective. We have gained new understanding of the algorithmic strengths and weaknesses of finite-state robots as they navigate square meshes. We have contrasted the powers of a single FSM vs. teams of ≥ 2 identical FSMs on three basic problems, each specified by a pair of fractions $\langle \varphi, \psi \rangle$ that specify the *anchor* tile $v^{(\varphi,\psi)}$ in any $n \times n$ mesh $\mathcal{M}_n$. Each anchor specifies a partition of $\mathcal{M}_n$ into quadrants. The *Anchor-Identification (A-I)* Problem has FSMs move to $v^{(\varphi,\psi)}$; the *Plot-Sweep (P-S)* Problem has FSMs sweep through each of the quadrants specified by $v^{(\varphi,\psi)}$; the *Compliance-Checking (C-C)* Problem has FSMs verify that every tile of $\mathcal{M}_n$ has a label that is unique to its quadrant. Single FSMs cannot solve the A-I or P-S Problems, but they can solve the C-C Problem; teams of ≥ 2 identical FSMs can solve all three problems, and they can speed up the C-C Problem linearly via parallelism. All problem solutions are *scalable:* a single FSM design works for all meshes. FSMs can sometimes use $\mathcal{M}_n$'s edges to *appear* to count to n, even though unbounded counting is impossible. The P-S and C-C Problems combine to show that single FSMs can sometimes *check* patterns that they are unable to generate.

6.2 Sample extensions. (1) One can generalize Theorem 3 to *sweep nonsquare submeshes;* e.g., if the "usher" FSM follows a diagonal trajectory, then the team sweeps a trapezoidal region. (2) One can *personalize the A-I Problem* so that FSM $\mathcal{F}^{(\varphi,\psi)}$ moves to the "copy" of tile $v^{(\varphi,\psi)}$ in $\mathcal{F}^{(\varphi,\psi)}$'s starting quadrant.

6.3 Prospective. The *Parking Problem* for FSMs in [17] focuses on the question "What can FSMs discover about where they reside within $\mathcal{M}_n$?" The current study, especially the A-I Problem, focuses on the question "How well can FSMs discover designated target tiles within $\mathcal{M}_n$?" An obvious goal for future research would be to extend the definitions of "where they reside" and "designated target tile." A valuable source of inspiration are robotic studies such as [1, 8, 11]. Another direction for the future would be to go beyond pure path planning/exploration by designing FSMs that can scalably find and transport "food" and that can avoid obstacles, inspired by, e.g., [2, 6, 8, 11, 15].

References

1. Adler, F., Gordon, D.: Information collection and spread by networks of patrolling ants. The American Naturalist 140, 373–400 (1992)
2. Basu, P., Redi, J.: Movement control algorithms for realization of fault-tolerant ad hoc robot networks. IEEE Network, 36–44 (July/August 2004)
3. Bender, M., Slonim, D.: The power of team exploration: two robots can learn unlabeled directed graphs. 35th IEEE Symp. on Foundations of Computer Science, 75–85 (1994)
4. Blum, M., Sakoda, W.: On the capability of finite automata in 2 and 3 dimensional space. 18th IEEE Symp. on Foundations of Computer Science, 147–161 (1977)
5. Budach, L.: On the solution of the labyrinth problem for finite automata. Elektronische Informationsverarbeitung und Kybernetik (EIK) 11(10-12), 661–672 (1975)
6. Chen, L., Xu, X., Chen, Y., He, P.: A novel FSM clustering algorithm based on Cellular automata. IEEE/WIC/ACM Int'l Conf. Intelligent Agent Technology (2004)
7. Cohen, R., Fraigniaud, P., Ilcinkas, D., Korman, A., Peleg, D.: Label-guided graph exploration by a finite automaton. ACM Trans. on Algorithms 4 (2008)
8. Geer, D.: Small robots team up to tackle large tasks. IEEE Distributed Systems Online 6(12) (2005)
9. Goles, E., Martinez, S. (eds.): *Cellular Automata and Complex Systems.* Kluwer, Amsterdam (1999)
10. Gruska, J., La Torre, S., Parente, M.: Optimal time and communication solutions of firing squad synchronization problems on square arrays, toruses and rings. In: Developments in Language Theory (C.S Calude, E. Calude, M.J. Dinneen, Eds.) Lecture Notes in Computer Science 3340, Springer, Heidelberg, 200–211 (2004)
11. http://www.kivasystems.com/
12. Koenig, S., Szymanski, B., Liu, Y.: Efficient and inefficient ant coverage methods. Annals of Mathematics and Artificial Intelligence 31, 41–76 (2001)
13. Marchese, F.: Cellular automata in robot path planning. EUROBOT'96, 116–125 (1996)
14. Müller, H.: Endliche Automaten und Labyrinthe. Elektronische Informationsverarbeitung und Kybernetik (EIK) 11(10-12), 661–672 (1975)
15. Rosenberg, A.L.: Cellular ANTomata: path planning and exploration in constrained geographical environments. Advances in Complex Systems, to appear (2012)
16. Rosenberg, A.L.: *The Pillars of Computation Theory: State, Encoding, Nondeterminism.* Universitext Series, Springer, Heidelberg (2009).
17. Rosenberg, A.L.: Ants in parking lots. 16th Int'l Conf. on Parallel Computing (EURO-PAR'10), Part II. In: Lecture Notes in Computer Science 6272, Springer, Heidelberg, 400–411 (2010)
18. Russell, R.: Heat trails as short-lived navigational markers for mobile robots. Int'l Conf. on Robotics and Automation, 3534–3539 (1997)
19. Spezzano, G., Talia, D.: The CARPET programming environment for solving scientific problems on parallel computers. Parallel and Distributed Computing Practices 1, 49–61 (1998)