

# Agent Knowledge and Beliefs in a Cloud

N.V. Shilov<sup>1,2</sup>(✉)

<sup>1</sup> A.P. Ershov Institute of Informatics Systems,

Lavren'ev avenue, 6, Novosibirsk, Russia

<sup>2</sup> Nazarbayev University, Kabanbay Batyr avenue, 53, Astana, Kazakhstan

shilov@iis.nsk.su, nikolay.shilov@nu.edu.kz

**Abstract.** Cloud computing is a concept that is in use since late 2000s related to consumption of distributed computer resources, namely servers and networks for data storage and access. In the paper we examine knowledge-based algorithms for agents that have access to a resource center to use some of available discrete resources. We assume that resource items are passive, they form a cloud, any item can be lend on demand to any agent if and only if there is no races for this item with other agents. All agents are rational and can communicate with each other in P2P-manner, negotiate, flip and swap (change intentions) so that all flips/swaps always must be rational for participating agents. The problem is to design a multiagent algorithm, which allows each agent sooner or later to access some resource item. We present a uniform algorithm scheme and then specialize for the following particular problems: *Robots in Space* and *Rational Agents at the Marketplace*.

## 1 Introduction

The issues of *trust aspects* (functional correctness, safety, security, reliability, credibility, usability) in multi-agent systems have received a lot of attention. Methods to guarantee functional correctness, safety, and security as well as techniques to ensure reliability in distributed, self-organizing systems are under investigation by different research communities and (in particular) in a multiagent research paradigm.

In general, a paradigm is an approach to the formulation (formalization) of problems and the ways to solve them. The term comes from Greek and means pattern, example. A contemporary meaning of science paradigm is due to well-known book [10] by T. Kuhn. Robert Floyd was the first who had explicitly used the term paradigm in the Computer Science context in his Turing Award Lecture in 1978.

Multiagent paradigm is a common name for several related research and development approaches in Computer Science, in Artificial Intelligence, Information Systems, etc. In this paper we are bound to Computer Science multiagent paradigm as sketched below.

---

The research has been supported by Russian Foundation for Basic Research (grant 13-01-00643-a).

A *distributed system* consists of multiple autonomous “computers” (programs with distributed memory) that communicate through a network [20]. A *multi-agent system* is a distributed system that consists of asynchronous (*rational*) *agents*. An agent is an autonomous reactive and projective *object* (in OO-sense) whose internal states may be characterized in terms of *Beliefs* (B), *Desires* (D), and *Intentions* (I). (Agents of the described kind are usually called BDI-agents [22].) A rational agent has clear “preferences” and always chooses the action (in feasible actions) that leads to the “best” outcome for itself; in contrast, a bounded rationality is “decision making” limited by the cognitive and deductive abilities of agents or other constraints (e.g. amount of time they have to make decisions). A *multiagent algorithm* is a distributed algorithm (protocol) [21] that solves some problem by means of cooperative work of agents in a multiagent system.

Agent’s beliefs represent its ideas and opinions about itself, other agents, and the network; these ideas and opinions may be incomplete, inconsistent, and (even) incorrect in contrast to agent *knowledge*. We distinguish belief and knowledge notions according to the famous Plato thesis: *Knowledge is true belief*. Thus our approach to *knowledge* and *belief* is not very formal like in [6], but (we demonstrate in the paper that) it can be formalized in terms of *interpreted systems* [4, 11, 19].

Agent’s desires represent its long-term aims, obligations and purposes (that may be controversial). Agent’s intension is its plan how to implement its desires or a short-term individual planning. *Reactivity* means that every agent could change its beliefs, desires, and intentions after communication and interaction with other agents (the *environment*). (In particular, some former beliefs may transfer to knowledge about the environment, some may be refuted.) *Projectivity* is agent’s ability to design/modify/adopt short-term plans (i.e. its intentions) according to updated information about the environment.

Some other notions related to multiagent systems are defined below. A *autonomous* agent changes its personal beliefs, desires and intentions by its own reasons, the change can’t be decreed by any other agent. A *rational* agent has clear preferences and always chooses the action (in feasible actions) that leads to the best outcome for itself; a *bounded rationality* is decision making (basically planning) limited by the cognitive abilities of agents (e.g. the finite amount of time they have to make decisions) [14]. Agent’s *privacy* is an opportunity to hide in negotiations data that the agent supposes to be private. [5, 7]; in contrast, agent’s *anonymity* is its opportunity not to revile its identity in negotiations.

In this paper we study multiagent algorithms (protocols for multiagent systems) for solving instances of the following general problem that we call *Discreet Resources Allocation Problem* (DRAP):

There is a cloud of discreet resources consisting of a fix number of items (or pieces that are not dividable any more); there are also rational agents, each of which pretends for individual exclusive access to an item (exactly one); all agents can communicate and negotiate in peer-to-peer manner; the problem is to define a belief- and rationality-based protocol of

pairwise communications and negotiations, virtual flips/swaps of intentions (i.e. items) between agents, a protocol that eventually leads every agent to the knowledge about the individual resource item that belongs to the agent; it will be an advantage if the resulting global resource allocation will meet some global optimality criterion.

One can argue that DRAP can be solved by distributed *consensus* or *leader election* algorithms [3]. It is true in a distributed paradigm, but not in the multiagent paradigm, because sharing of private data is very legal in distributed systems, but not in multiagent systems. In particular, it is possible to elect a leader, pass it agents' private data, then let the leader to assign individual resource items to all agents according to some global optimality criterion. But agents in that distributed algorithm are too passive in problem solving and they send too much individual data to the leader. More over, in multiagent paradigm every agent (including the leader) should care just about itself (i.e. its individual knowledge about its individual safety and its individual resource item), but not about other agents and the system.

Doctoral Dissertation [12] gives a general survey of the Distributed Resource Allocation Problem in a frame of agent-based approach. Using standard terminology we could distinguish the following features of our approach to the problem.

- Resources are discreet, not dividable, not sharable, static, single-unit.
- Agents have quantitative preference structure, i.e. agents' utility function maps numbers to a set of resources.
- Resource allocations are evaluated by the utilitarian welfare which is the sum of individual utilities.
- Our multi-agent systems handle homogeneous populations, where all agents act according to the same behavior scenario.
- The social graph (i.e. agents' communication graph) is complete: everybody can talk with everyone.
- Any time each agent knows its own intention, but never the hole intentions of all other agents.
- We use bilateral swap transactions (which involve only two agents at a time) and individual flips.

In addition our algorithms are knowledge- and rational-based, privacy-preserving but not anonymous (i.e. agents need to revile their identity).

The rest of the paper is organized as follows. In the next Sect. 2 we introduce and discus particular examples of DRAP. Then, in Sect. 3, we present and discuss the algorithm for DRAP and how to specialise it for two paerticular instances of DRAP (from Sect. 2). In the concluding Sect. 4 we discuss relations of the presented research to so called *social computing* [1] and/or *social software* [13].

## 2 DRAP Special Cases

### 2.1 RinS: Robots in a Space

RinS problem formulated below is a multidimensional generalization of *Mars Robot Puzzle* (MRP) that has been studied in [2,17]: MRP is just RinS in 2-dimensional Euclidian space.

There are  $n > 0$  autonomous robots and the same number of shelters in a general position in a Euclidean space. A position of every shelter is fixed and known to every robot. Every robot knows about existence of all other robots, but does not know their locations. Robots can communicate in peer-to-peer manner only. At some moment each robot stops and fixes its current position. Then every robot should choose a shelter to move in by a straight way. Robots should not collide. The problem: Design a multiagent algorithm that guarantees that every robot will eventually know an individual shelter such that its straight route to the selected shelter never intersects with straight routes of other robots.

We would like to point out that RinS and MRP are related to Multi-Agent Programming Contest (MAPC, <http://multiagentcontest.org>). In 2011 organizers of MAPC *began the fourth phase with the definition of a new scenario: “Agents on Mars”. The goal is to implement a team of cooperating agents with different roles in order to occupy zones on planet Mars. The challenge of the scenario is its increased complexity, that is that we have defined 5 roles of agents with different properties and capabilities.* But the difference between RinS/MRP and MAPC is crucial: we care about straight routes and formal correctness.

### 2.2 RAM: Rational Agents at the Marketplace

RAM problem formulated below has been discussed first in [18].

At the Marketplace there are  $n > 0$  buyers and  $m \geq n$  salesman. Every salesman sells a (single) unit of some indivisible good (“a piece of cake”) by individual prices for different buyers. Salesmen are passive: they do not care if their goods are sold out and who buy the goods. But every buyer is rational: it has to purchase exactly one unit of goods and it knows its individual prices from every salesman. Buyers can choose and flip salesmen, negotiate pairwise, swap salesmen in pairs, make price concessions. But every above action should benefit a buyer. A buyer can make a deal with a salesman iff it knows that nobody pretends for a deal with this salesman. The problem: Design a multiagent algorithm for buyers which guarantees that every buyer eventually buy something.

The difference between RAM and RinS/MRP is manifold. First, in RAM agents are assumed to be rational, while in MRP agents do not care about their benefits (preferences) at all. Next, MRP has a clear geometric interpretation,

but it is not clear from the very beginning, whether any intersection-free set (of routes) exists, and, hence it is not obvious that a desired protocol may exist. Fortunately, the existence of these routes could be proved by contradiction, or by reduction to the *assignment problem* in Graph Theory, or to the convex hull problem in Combinatorial Geometry.

The RAM problem is related to the classic *Cake Cutting Problem* (CC-problem, also known as *Fair Division Problem*) that has been introduced by a group of Polish mathematicians, H. Steinhaus, B. Knaster and S. Banach [1]. The CC-problem is to divide an infinitely dividable resource (“cake”) in such a way that all recipients believe that they have received a fair amount. A special cases of the problem are *proportional* and *envy-free* division. Differences between RAM and CC-problem are evident: in CC-problem a cake is an infinitely dividable resource, while in RAM-problem a “resource” has been cut already; solutions of the CC-problem may be sequential, while solutions (if any) of RAM must be multiagent (i.e. distributed, parallel and concurrent) by the problem statement.

At the same time RAM problem is also closely related to the following *Stable Marriage Problem* (SMP) [9]. Given  $n$  men and  $n$  women, where each person has ranked all members of the opposite sex with a unique number between 1 and  $n$  in order of preference, marry the men and women together such that there are no two people of opposite sex who would both rather have each other than their current partners. (If there are no such pairs, all the marriages are said to be stable.) A non-deterministic but centralized and sequential algorithm has been developed by D. Gale and L. Shapley for SMP [9]. It makes the difference between SMP and RAM: SMP implicitly assumes a single sequential matchmaker, while RAM explicitly states that the problem must be solved by agents themselves.

### 2.3 RaceP: Race for Processors

The following new problem is closely related to DARP but is not a special case of the problem due to *dynamics* of arriving and served processes.

There is a resource center consisting of (1)  $n > 0$  (different) processors, (2) a pool with  $m > 0$  tasks, and (3) a monitor for processor’s load and a state of the processors. The monitor have marked all tasks in the pool by a time stamp upon their arrival (into the pool); assigns instantly a free processor to some task, if the task requests the processor; removes assigned tasks from the pool, and returns processors to the center after task execution. Every task is an agent, that knows an individual “wanted” process. The problem: Design a multiagent algorithm for tasks which guarantees that every task in the pool eventually be assigned by a processor to be performed.

## 3 Algorithm and Correctness

An *unified* multiagent algorithm SOpt (*Search of Optimum*) for DRAP is presented below in this section. It is a generalization of SMEx-algorithm [2, 17] that

solves MRP. Correctness of specialized algorithms for RinS/MRP and RAM problems have been proved in [2, 17], but its adjustment for RaceP is still an open question. The correctness proofs follow main steps and are similar to the proof of SMEx algorithm. Both protocols SMEx and SOpt rely upon the following fairness communication assumption [17]: communication (in a multiagent system) is said to be *fair*, if every agent which would like to communicate with any other agent will communicate eventually. Fairness has been discussed in [8].

Algorithm SOpt for an individual agent can be described informally as follows. Any time every agent can ask a monitor about the set of *Agents* that currently compete for resources. At every moment every agent has some particular resource item as its current intention; at the very beginning this intention is defined by function  $INI : Agents \rightarrow Resources$ . Conflicts between agents can be checked by predicate  $Conf : Agents \times Resources \times Agents \times Resources \rightarrow Boolean$  such that for all agents  $i \neq j$ , for all resource items  $u_i$  and  $u_j$ ,  $Conf(i, u_i, j, u_j) \Leftrightarrow Conf(j, u_j, i, u_i)$ . Beliefs of every agent are represented by several integer counters:

- $NC$  for *Number of Conflicts* is used by instant and time agents,
- $CF$  for *Conflict-Free* agents is used by time agents only.

We distinguish also *instant* and *time* agents: the former agent can catch (or grab) the allocated piece immediately and instantly, the latter needs some time to catch the allocated piece: an instant agent grabs its resource instantly as soon  $NC = 0$ , while a time agent waits for all other agents, since acquiring of the resource needs time (during which a new conflict may occur).  $NC$  represents agent's upper estimation of number of agents with whom it may have conflict, and, respectively,  $CF$  represents its lower estimation of number of agents that have no conflicts at all. In particular,

- the agent believes that it does not conflict with any other agent as soon as  $NC = 0$ ;
- the agent believes that there is no conflicts in the system as soon as  $NC = 0$  and  $CF = 2 \times (n - 1)$ , i.e. it believes that it has no rivals, and it checks *twice* that all other agents believe that they do not have conflicts also.

But in the case when two agents have a conflict, then they resolve the trouble by non-deterministic function  $Sol : Agents \times Resources \times Agents \times Resources \rightarrow Resources$  such that for all agents  $i \neq j$ , for all resource units  $u_i$  and  $u_j$ , if  $Conf(i, u_i, j, u_j)$  then  $Sol(i, u_i, j, u_j)$  and  $Sol(j, u_j, i, u_i)$  are different resource unites.

Pseudocode of the SOpt-algorithm follows, but first we would like to comment a meaning of some variables: **Me** is a variable for personal agent's identification number; **cur.un** and **par.un** are the variables for intentions of the agent and its partner (i.e. for resource items); **par.bel** is a variable for partner's belief that the partner is conflict-free; **contacts** is a variable for a set of agents. (The algorithm is given for time agents. Please, ignore all instances of  $CF$  counter for instant agents.)

```

algorithm SOpt::
const Me: integer in [1..n]
var NC: integer in [1..(n-1)];
var CF: integer in [1..(n-1)];
var contacts: set of [1..n];
var partner: integer in [1..n];
var cur_un, par_un: integer in [1..n];
var par_bel: boolean;
begin
1:  NC:= (n - 1); CF:= 0;
2:  cur_un:= INI(Me);
3:  repeat
4:    if NC > 0 then NC:= (n - 1);
5:    contacts:= Agents \ {Me};
6:    repeat
7:      partner:= any agent in the contacts ready to communicate1;
8:      start communication session with the partner:
9:      { send (<cur_un>;<(NC=0)?>) to partner ||
10:       receive (<par_un>;<par_bel>) from partner}
11:     if Conf(Me, cur_un, partner, par_un) then
12:       {cur_un:= Sol(Me, cur_un, partner, par_un);
13:        NC:= (n - 1); CF:= 0}
14:     else if NC > 0
15:       then {NC:= (NC - 1); CF:= 0}
16:       else if par_bel
17:         then CF:= CF + 1
18:         else {NC:= (n - 1); CF:= 0}
19:     close communication session with partner;
20:     contacts:= remove partner from contacts;
21:   until contacts becomes empty
22: until (NC = 0  $\wedge$  CF = 2×(n - 1))
end.

```

**Proposition 1.** *If a multiagent system consists of a fair communication scheduler and agents that all execute SOpt protocol, and system terminates, then all agents will know their conflict-free pieces of the resource upon the termination.*

A *well-founded set* is a partial order  $(D, \leq)$  without infinite (strictly) decreasing sequences. Let us say that a distributed system is well-founded, if there exists a well-founded set  $(D, \leq)$  and a *well-mapping*  $F$  from system configurations (i.e. its global states) into  $D$  such that for all agents  $i \neq j$ , for all resource units  $u_i$  and  $u_j$ ,  $\text{Conf}(i, u_i, j, u_j)$  implies that execution of  $\text{Sol}(i, u_i, j, u_j)$  and  $\text{Sol}(j, u_j, i, u_i)$  in any order reduces the value of  $F$ .

---

<sup>1</sup> A scheduler resolves this request.

**Proposition 2.** *If a multiagent system consists of a fair communication scheduler and agents that all execute  $SOpt$  protocol, and the system is well-founded then the system eventually terminates.*

The specialization of above general algorithm for the particular problems RinS and RAM consists in a manner of recognizing and resolving conflicts. In case of RinS

- predicate  $Conf$  is true iff routes of robots intersects,
- function  $Sol$  swaps shelters for robots, and
- well-mapping  $F$  maps system configurations into the sum of all distances from robots to shelters.

In case of RAM

- predicate  $Conf$  is true iff there is a competition of buyers for the same salesman,
- function  $Sol$  implements a game with better price as a gain, and
- well-mapping  $F$  maps system configurations into a total price for all goods.

## 4 Conclusion

### 4.1 Summary

In the paper we discussed multiagent approach to the discrete resource allocation problem (DRAP), presented a multiagent algorithm that solves the problem under assumption of communication fairness and well-foundedness, and presented particular examples of problems that can be solved on base of the our results.

We also have to remark that we considered very idealistic multiagent systems that consists of absolutely reliable (non-faulty) agents in a static situation. Hence a topic for further research may be study of more realistic case of faulty agents and dynamic systems. Anonymity could be studied also, since our  $SOpt$  algorithm requires from agents to inform partners about their identity.

There exist sophisticated distributed algorithms for resource allocation problem, but *formal* functional correctness and etc. usually is out of scope of these works. In contrast, a contribution of our paper is to study of this aspect (although for idealized examples). We have to mention in this extended abstract that our correctness proof is manual. But one can observe that the formal description of the interpreted system for DRAP makes possible to verify the algorithm automatically with aid of techniques developed in [16]. It will be a topic for further research also.

### 4.2 Cloud vs. Crowd

We have mentioned in Sect. 2 that Ratioanl Agents at Marketplace (RAM) problem is closely related to the classic Cake Cutting Problem (CC-problem) and discussed similarity and differences between these two problems. But in spite of



these differences, RAM and CC problems have something in common since they both are examples of a new research paradigm of that is called *social computing* or *social software* [1, 13]. The essence of this paradigm is sketched in the next paragraph.

In the modern world very many social requirements and procedures have algorithmic character. These requirements can be written as (semi-)formal specifications and procedures — software (in a pseudo-code). Then the properties of these procedures can be analyzed and verified by formal methods. Well, the results of the formal analysis or verification may be interpreted in socially significant terms. And though about social computing/software started talking only in a XXI century, but it is possible to consider as the first example of application of this paradigm research of the Cake Cutting Problem by H. Shteinhaus, B. Knaster and S. Banach.

In particular, Rational Agents at Marketplace can be considered as an example of social computing/software research [15]. In social computing/software paradigm “crowd” with communication network and access to resources may be considered as agents and cloud of resources. But in this context study of efficiency, fairness, privacy and anonymity of multiagent algorithms get much more importance than in this paper.

## References

1. Brams, S.J., Taylor, A.T.: Fair Division - From Cake-Cutting to Dispute Resolution. Cambridge University Press, Cambridge (1996)
2. Bodin, E.V., Garanina, N.O., Shilov, N.V., Mars, N.V.: Robot puzzle (a multiagent approach to the Dijkstra problem). Model. Anal. Inf. Syst. **18**(2), 113–128 (2011). (in Russian)
3. Cachin, C., Guerraoui, L.S.: Rodrigues Introduction to Reliable and Secure Distributed Programming, 2nd edn. Springer, New York (2011)
4. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning About Knowledge. MIT Press, Cambridge (1995)
5. Halpern, J., O'Neill, K.: Anonymity and information hiding in multiagent systems. J. Comput. Secur. **13**(3), 483–514 (2005)
6. Hintikka, J.: Knowledge and Belief. Cornell University Press, Ithaca (1962)
7. Hughes, D., Shmatikov, V.: Information hiding, anonymity and privacy: a modular approach. J. Comput. Secur. **12**(1), 3–36 (2004)
8. de Jong, S., Tuyls, K., Verbeeck, K.: Fairness in multiagent systems. Knowl. Eng. Rev. **23**(2), 153–180 (2008)
9. Knuth, D.E.: Stable Marriage and its Relation to Other Combinatorial Problems. CRM Proceedings and Lecture Notes, vol. 10. American Mathematical Society, Providence (1997)
10. Kuhn, T.S.: The Structure of Scientific Revolutions, 3rd edn. University of Chicago Press, Chicago (1996)
11. Lomuscio, A., Ryan, M.D.: On the relation between interpreted systems and Kripke models. In: Wobcke, W., Pagnucco, M., Zhang, C. (eds.) Agents and Multi-Agent Systems Formalisms, Methodologies, and Applications. LNCS (LNAI), vol. 1441, pp. 46–59. Springer, Heidelberg (1997)

12. Nongaillard, A.: An Agent-Based Approach for Distributed Resource Allocations. Doctoral dissertation. Concordia University Montreal, Canada (2009)
13. Parikh, R.: Social software. *Synthese* **132**, 187–211 (2002)
14. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall, Saddle River (2010)
15. Satekbayeva, A., Shilov, N.V.: Some results on multiagent algorithms in social computing/software context. *Information* **17**(1), 229–240 (2014)
16. Shilov, N.V., Garanina, N.O., Choe, K.-M.: Update and abstraction in model checking of knowledge and branching time. *Fundamenta Informaticae* **72**(1–3), 347–361 (2006)
17. Shilov, N.V., Garanina, N.O., Bodin, E.V.: Multiagent approach to a Dijkstra problem. In: Proceedings of Workshop on Concurrency, Specification, and Programming CS&P 2010, pp. 73–84. Humboldt-Universität zu, Berlin (2010)
18. Shilov, N.V., Garanina, N.O.: Rational agents at the marketplace. In: Proceedings of Workshop on Concurrency, Specification and Programming CS&P 2011, pp. 465–476. Bialystok University of Technology, Pułtusk, Poland, 28–30 September 2011
19. Su, K., Luo, X., Sattar, A., Orgun, M.A.: The interpreted system model of knowledge, belief, desire and intention. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), pp. 220–222 (2006)
20. Tanenbaum, A.S., Van Steen, M.: Distributed Systems: Principles and Paradigms, 2nd edn. Prentice Hall, Saddle River (2007)
21. Tel, G.: Introduction to Distributed Algorithms, 2nd edn. Cambridge University Press, Cambridge (2000)
22. Wooldridge, M.: An Introduction to Multiagent Systems. Jhon Wiley & Sons, Chichester (2002)

<http://www.springer.com/978-3-319-25042-7>

Embracing Global Computing in Emerging Economies

First Workshop, EGC 2015, Almaty, Kazakhstan,

February 26-28, 2015. Proceedings

Horne, R. (Ed.)

2015, X, 149 p. 52 illus. in color., Softcover

ISBN: 978-3-319-25042-7