J. Parallel Distrib. Comput. 75 (2015) 67–80

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Energy-aware parallel self-reconfiguration for chains microrobot networks

Hicham Lakhlef [a,*], Julien Bourgeois [a], Hakim Mabed [a], Seth Copen Goldstein [b]

[a] UFC/FEMTO-ST, UMR CNRS 6174, 1 cours Leprince-Ringuet, 25201 Montbeliard, France

[b] School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA

HIGHLIGHTS

- We present a parallel and energy-ware protocol for self-reconfiguration of microrobots.
- From chains configurations to squares configurations.
- This protocol is efficient and scalable because it is map-less.
- This protocol needs a constant complexity of memory usage.
- Evaluation is made with the Dynamic Physical Rendering Simulator.

ARTICLE INFO

Article history:
Received 1 December 2013
Received in revised form
25 September 2014
Accepted 2 October 2014
Available online 17 October 2014

Keywords:
MEMS microrobot
Distributed algorithm
Parallel algorithm
Self-reconfiguration
Logical topology
Energy

ABSTRACT

MEMS microrobots are miniaturized electro-mechanical elements, made using the techniques of micro-fabrication. They have limited energy capacity and low memory space. Self-reconfiguration is required for MEMS microrobots to complete their mission and/or to optimize their communication. In this paper, we present a self-reconfiguration protocol from a straight chain to square organization, which deals with MEMS microrobots characteristics. In the proposed protocol, nodes do not have the map of their target positions which makes the protocol portable, standalone, and the memory complexity is bounded by a constant. This paper improves a former solution by using parallelism in the movements of microrobots to optimize the time and the number of movements and by making the algorithm energy-aware. So each node is aware of the amount of energy that it will spend, which will improve the energy consumption. Our algorithm is implemented in Meld, a declarative language, and executed in a real environment simulator called DPRSim.

1. Introduction

Micro electro mechanical system (MEMS) is a technology that enables the batch fabrication of miniature mechanical structures, devices, and systems. MEMS are miniaturized and low-power devices that can sense and act. It is expected that these small devices, referred to as MEMS nodes, will be mass-produced, making their production cost almost negligible [8]. Their applications will require a massive deployment of nodes, thousands or even millions [36] which will give birth to the concept of Distributed Intelligent MEMS (DiMEMS) [4].

The size of MEMS nodes differs from well below one micron to few millimeters. A DiMEMS device is composed of typically hundreds of MEMS nodes. Some DiMEMS devices are composed of mobile MEMS nodes [1], some others are partially mobile [9] whereas others are not mobile at all [4]. Due to their small size and the batch-fabrication process, MEMS microrobots are potentially very cheap, particularly through their use in many areas in our lifetime [10].

One of the major challenges in developing a microrobot is to achieve a precise movement to reach the destination position while using a very limited power supply. Many different solutions have been studied for example, within the *Claytronics* project [1,2,7,25] each microrobot can only turn around its neighbor which introduce the idea of a collaborative way of moving. But, even if the power requested for moving has been lowered, it still costs a lot regarding the communication and computation requirements.

* Corresponding author.
E-mail addresses: hlakhlef@femto-st.fr (H. Lakhlef), julien.bourgeois@femto-st.fr (J. Bourgeois), hmabed@femto-st.fr (H. Mabed), seth@cs.cmu.edu (S.C. Goldstein).

http://dx.doi.org/10.1016/j.jpdc.2014.10.003
0743-7315/

Optimizing the number of movements of microrobots is therefore crucial in order to save energy [14].

MEMS microrobots topic is gaining an increasing attention since large-scale swarms of robots will be able to perform several missions and tasks in a wide range of applications such as odor localization, firefighting, medical service, surveillance, search, rescue, and security [8]. The self-reconfiguration for MEMS microrobots is necessary to do these tasks. In the literature, self-reconfiguration can be seen from two different points of view. On the one hand, it can be defined as a protocol, centralized or distributed, which transforms a set of nodes to reach the optimal logical topology from a physical topology [11]. For example, if we have a connected chain of n microrobots then the complexity of message exchange if a node broadcasts a message to others will be $O(n)$ in the worst case. If we reconfigure the chain to a square the complexity will be $O(\sqrt{n})$ in the worst case. On the other hand, the self-reconfiguration is built from modules which are autonomously able to change the way they are connected, thus changing the overall shape of the logical network [7,31]. This process is difficult to control, because it involves the distributed coordination of a large number of identical modules connected in time-varying ways. The range of exchanged information and the amount of displacement, determine the communication and energy complexity of the distributed algorithm. When the information exchange involves close neighbors, the complexity is moderate and the resulting distributed self-reconfiguration scales gracefully with network size.

An open issue is whether distributed self-reconfiguration would result in an optimal configuration with a moderate complexity in message, execution time, number of movements and memory usage.

As said before, MEMS microrobots are low-power and low-memory capacity devices that can sense and act. A solution of self-reconfiguration should deal with MEMS microrobots characteristics. Self-reconfiguration with shared map does not scale. Because the map (predefined position of the target shape) consists of P positions and each node must have a memory capacity, at least, of P positions. Therefore, if P is very high, the self-reconfiguration will be not feasible. In this paper, we present an energy-aware parallel reconfiguration algorithm, without predefined positions of the target shape, which reduces memory usage to $O(1)$. This algorithm ensures the networks connectivity throughout all its execution time. This work takes place within the Claytronics project and aims at optimizing the logical topology of the network through rearrangement of the physical topology as we will see in the next sections.

2. Related works

Many terms refer to the concept of self-reconfiguration. In several works on wireless networks the term used is *self-organization*. This term is also used to express the partitioning and clustering of ad-hoc networks or wireless networks to groups called cliques or clusters. Also, the self-organization term can be found in protocols for sensor networks to form a sphere or a polygon from a center node [23,24,39]. The term *redeployment* is also a new term to address self-reconfiguration for sensor networks [16,13,29]. For self-reconfiguration with robots or microrobots, there are the protocols [31,32,15] where the desired configuration is grown from an initial seed module. A generator uses a 3D model of the target configuration and outputs a set of overlapping blocks which represent this configuration. In the second step, this representation is combined with a control algorithm to produce the final self-reconfiguration algorithm. In [38], the authors propose map based distributed algorithm for self-reconfiguration of modular robots from arbitrary to straight chain configuration.

A growing number of research on self-reconfiguration for microrobots have used centralized algorithms, among them we find centralized self-assembly algorithms [27]. Other approaches give each node a unique ID and a predefined position in the final structure [37]. The drawback of these methods is the centralized paradigm and the need for nodes identification. More distributed approaches that need the map of the target shape in [12,5,28,22]. In simulation, the authors in [30,32] have demonstrated algorithms for self-reconfiguration and directed growth of cubic units based on gradients and cellular automata. The authors in [3] have shown how a simulated modular robot (Proteo) can self-configure into useful and emergent morphologies when the individual modules use local sensing and local control rules.

In [35] the authors developed a centralized algorithm for reconfiguration (with predefined positions of the target chain) of an initial chain configuration into another chain configuration and then from a straight chain into an arbitrary goal that fulfills certain admissibility requirements [34]. The distributed version of this algorithm was given in [38]. Recent work in [21] demonstrated a time complexity of $O(n)$ for probabilistic reconfiguration of large systems of hexagonal metamorphic robots for single-move algorithms, in which at most one module can move in a time step.

Claytronics, is the name of a project led by Carnegie Mellon University and Intel corporation. In Claytronics, microrobots called catoms (Claytronics atoms) are assembled to form larger objects. The idea is to have hundreds of thousands of microrobots forming objects of any shape. Like the cells in a body or in a complex organism, each small member of the whole is committed to doing its own part and communication between microrobots helps in building the final shape.

Many works have already been done within the Claytronics project. In [6], the authors propose a metamodel for the reconfiguration of catoms starting from an initial configuration to achieve a desired configuration using *creation* and *destruction* primitives. The authors use these two functions to simplify the movement of each catom. In [7], the authors present a scalable distributed reconfiguration algorithm with the Hierarchical Median Decomposition, to achieve arbitrary target configurations without a global communication. Another scalable algorithm has been presented in [25]. In [2], a scalable protocol for Catoms self-reconfiguration is proposed, written with the Meld language [1,26] and using the creation and destruction primitives. In all these works, the authors assume that all Catoms know the correct positions composing the target shape at the beginning of the algorithm and each node is aware of its current position. The first self-reconfiguration without predefined positions of the target shape appears in [17]. However, this solution is not parallelized, is not energy-aware and takes longer to achieve the reconfiguration. In this former solution, the choice of the initiator node is simple, but this initiator is not the best one to get a good execution time for the self-reconfiguration protocol. Also, controlling the movements of nodes was simple compared to this new solution because only three states were required. Nevertheless, the solutions in [17,18] take longer to achieve the reconfiguration protocol. Therefore, they are time-consuming, and do not maximize the lifetime of nodes. To cope with these disadvantages, we introduced the use of the parallelism in movements. Within this new solution, and because of the parallelism in movements, the control has become more complex, as new states are required to deal with a lot of cases to handle the parallelism in order to make it optimal. We presented in [19] an algorithm of reconfiguration from any starting physical topology to a square, this algorithm does not ensure the connectivity of the network during the reconfiguration.