Expert Systems with Applications 39 (2012) 3817–3831

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

Multi-robot path planning using co-evolutionary genetic programming

Rahul Kala [*]

School of Cybernetics, School of Systems Engineering, University of Reading, Whiteknights, Reading, Berkshire, UK

ARTICLE INFO

Keywords:
Path planning
Motion planning
Mobile robotics
Genetic programming
Grammatical evolution
Co-operative evolution
Multi-robot systems

ABSTRACT

Motion planning for multiple mobile robots must ensure the optimality of the path of each and every robot, as well as overall path optimality, which requires cooperation amongst robots. The paper proposes a solution to the problem, considering different source and goal of each robot. Each robot uses a grammar based genetic programming for figuring the optimal path in a maze-like map, while a master evolutionary algorithm caters to the needs of overall path optimality. Co-operation amongst the individual robots' evolutionary algorithms ensures generation of overall optimal paths. The other feature of the algorithm includes local optimization using memory based lookup where optimal paths between various crosses in map are stored and regularly updated. Feature called wait for robot is used in place of conventionally used priority based techniques. Experiments are carried out with a number of maps, scenarios, and different robotic speeds. Experimental results confirm the usefulness of the algorithm in a variety of scenarios.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

The problem of multi-robot motion planning deals with computation of paths of various robots such that each robot has an optimal or near optimal path, but the overall path of all the robots combined is optimal. This is a more complex task as compared to a single-robot motion planning, where the factor of coordination among the various robots is not applicable, and the single robot can use its own means to compute the path (Parker, Schneider, & Schultz, 2005). The problem of motion planning may be centralized or decentralized. In centralized planning all the robots are centrally planned by a planner, usually taking into account all the complex interactions that they may have. This results in the generation of a very complex configuration space, over which the search is to be performed. The decentralized planning, on the other hand, has an independent planner for every robot. Each robot is planned separately in its own configuration space, which makes the planning much simpler. Then efforts may be made to avoid the possibility of collision between the various robots. The centralized planning is more time consuming, but optimal as compared to decentralized planning (Arai & Ota, 1992; Sánchez-Ante & Latombe, 2002).

In this paper we assume a simple problem modeling scenario. We have a maze like map of $M \times N$ grids. Each grid may be black or white, denoting the presence or absence of obstacle, which are all assumed to be stationary. The grid is however not the unit position for robots which can occupy partial grid positions on the map. The complete map consists of horizontal and vertical lanes, which cris-cross each other at grids known as crosses. The task is to move n robots $R_1, R_2, \ldots, R_n$. Each robot R_i has a start grid S_i, which is its initial location; a goal grid G_i, which is the destination where it intends to arrive at the end of its journey; and a constant speed of motion V_i $(0 < V_i \leqslant 1)$ grids/unit time step. All robots are assumed to be of the same size of 1×1 grids. The planner needs to plan the path of all the robots R_i. Let the path of robot R_i be given by $P_i(t)$, which denotes its position at time t. We know that $P_i(0) = S_i$. Let the path generated be such that the robot R_i reaches the goal G_i at time T_i, i.e. $P_i(T_i) = G_i$. We assume that after reaching the goal, the robot disappears from the goal, and does not obstruct other robots to pass by, i.e. $P_i(T_i + 1) = \text{NIL}$. In this assumption we differ from the works over robot priorities by Oliver, Saptharishi, Dolan, Trebi-Ollennu, and Khosla (1999), Bennewitz, Burgard, and Thrun (2001, 2002), and Carpin and Pagello (2009). The planner needs to work such that the average time of travel for all the robots is as small as possible, i.e. $\min\left(\sum T_i\right)/n$. At the same time the planner needs to ensure that no collision occurs. Hence $P_i(t) \neq P_j(t)$, $1 \leqslant i,j \leqslant n$, $i \neq j$.

Genetic programming is extensively used evolutionary technique for problem solving. Here we represent the individual as a program and hence try to evolve it using the methodology of the evolutionary algorithms. The solution is obtained by executing the program so formed. Tree based representation of a program is a very common representation of individuals of genetic programming (Koza, 1992; Shukla, Tiwari, & Kala, 2010). The linear representation of the program of genetic programming gives rise to grammatical evolution. Here the individual is a sequence of integers. Every problem has its grammar, which represents the program syntax. The

---

[*] Tel.: +44 (0) 7424752843.
E-mail address: rkala001@gmail.com
URL: http://rkala.99k.org/

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.09.090

grammar is specified by Backus-Naur or BNF form. The generation of the phenotype solution from its genotype solution or sequence of integers takes place by selecting and firing rules specified in the grammar (O'Neill & Ryan, 2001, 2003). Our implementation of the genetic programming in this paper is similar to the general algorithm of grammatical evolution.

The evolutionary algorithms are used for solving various kinds of problems. These algorithms however face problems when the dimensionality of the problem becomes very large. In such a case the optimization provided by these algorithms becomes very slow, with fairly large chance of the algorithm getting struck at some local optima (Kala, Shukla, & Tiwari, 2010a). Co-operative evolutionary algorithms or co-evolutionary algorithms break up the problem into a number of smaller problems that together solve the main problems. The smaller problems have smaller dimensionality, and are hence very easy to be optimized. The different sub-problems or sub-modules evolve in parallel, which ultimately results in generation of very good modules that unite with each other to solve the problem well. Cooperation is encouraged between modules, and a module is given high fitness not only because it results in generation of higher fitness solutions, but also because it enables other modules to achieve high fitness value (Potter & De Jong, 1994, 2000; García-Pedrajas, Hervás-Martínez, & Muñoz Pérez, 2003).

In this paper we first present a co-evolutionary genetic programming based planning of multiple robots. Each of the individuals has its own optimization process that is based on the principles of grammatical evolution. These all try to generate and optimize the paths of the individual robots. However these need to consider the possibility of collision between the robots, for which the factor of co-operation comes into play and has been induced in the algorithm. A master genetic algorithm runs to select the best paths of the robots, out of all the paths available in the individual optimizations.

The major problem with the assumed scenario is the variable speed. This puts forward a number of scenarios, where sub-optimality may be possible without careful planning. Consider a big corridor with two robots A and B marching towards it. Consider maximum velocity of A being higher, i.e. $V_A \gg V_B$. Consider that robot B is closer to the corridor entry, and would enter it before A. But if robot B enters the corridor first, robot A would have to follow it, and both robots would be able to walk with a maximum velocity of V_B. It may be clearly seen that optimal strategy would be in case B waits for robot A to enter the corridor, and follows it inside the corridor. In such a case both robots would get a chance to travel by their maximum velocities. Hence we implement the concept of wait for robot, where a robot may be asked to wait so that some other robot may cross.

Optimization by evolutionary algorithms may get very complex in problems having too many dimensions and multiple modalities. In such a case it is often useful to use a local search strategy based on predefined heuristics that can aid the evolutionary algorithms. This saves the algorithm from likely convergence into local optima. The local search algorithm carries forward the task to converge the algorithm into some local or global optima, while the evolutionary algorithm does the task of locating the best possible regions, where the global optima may lie. In such a case the combined efforts of heuristics and evolutionary algorithms results in better optimization (Kala, Shukla, & Tiwari, 2009). The same problem of complex optimization is prevalent in the present algorithm. Here the evolutionary algorithm tries to optimize the individual robotic path. In such a case also we use a local search strategy which tries to modify the path to make it smaller in length. Longer paths are replaced by smaller paths. For carrying forward this replacement, an external memory is reserved that stores the smallest paths between all pairs of crosses. This table is regularly updated as more paths are found.

The novelty of the paper is threefold. We first propose a co-evolutionary genetic programming model for solving the problem. This model uses principles of cooperative evolution to generate paths that are optimal in time, by mixing centralized as well as decentralized planning. Secondly, we propose the model and issues with multi-speed, map based (single-lane) multi-robot motion planning. This includes the use of the wait for robot feature incorporated into the evolutionary approach. Thirdly we propose use of external memory for local optimization of the paths. This memory is initialized, updated, and queried for reduction in paths of individual robots.

The paper is organized as follows. In Section 2 we present some of the related works into the field. In Section 3 we present the co-evolutionary genetic programming model of the algorithm. Section 4 presents the concept of wait for robot. Section 5 presents the memory based local search used in the algorithm. The experimental results are given in Section 6. Section 7 gives the conclusion remarks.

2. Related works

The entire paradigm of path planning of multi-robots involve a large set of issues that range from dynamic changes in the robotic plan, to validity of non-holonomic constants, and execution and memory time complexity. Oliver et al. (1999) give a general architecture of path planning in these scalable scenarios. They followed a decentralized approach of planning where every robot does its own independent path planning. The map is built centrally at start and all changes are continuously monitored and any needed correction is made. Possible collisions are resolved by the coordination module, that uses a priority based coordination with a higher priority robot allowed to make the collision prone move, while the other robot compromises or waits. Possible collisions are predicted be extrapolation of the motion of robots, and any possibility is marked by a check-point. One of the robots is allowed to move, while the path of other may be re-planned.

Motion planning may be regarded as a part of the entire robotic architecture, for which again numerous models have been built. Asama, Matsumoto, and Ishida (1989) present a complete system for multi-robot planning, coordination, and control in their designed system ACTRESS (ACTor-based Robots and Equipments Synthetic System). In this system the authors demonstrate the mechanisms in which the different modules including sensors, processing, planning, manipulators, communication etc. come into play for intelligent and autonomous task accomplishment. The extension of their work can be found in Asama et al. (1991) where the authors developed a multi-level path planning. The planning was different for static and dynamic cases. Special preventions were taken for deadlock avoidance that may take place many times in planning of multiple robots. The complete hierarchy of the system includes static path planning, use of local algorithm, use of mobile rules, task prioritization, deadlock avoidance by low-level deadlock solver, deadlock avoidance by high-level deadlock solver, and human operator expertise. Vendrell, Mellado, and Crespo (2001) presented a general framework for motion planning with multiple robots. They highlighted the need to plan, sense, and accordingly re-plan by every robot. The authors further present a hierarchical model of the various robot activities that range from high abstraction to low abstraction. The hierarchy includes mission, task, action, motions, trajectories, and robot order.

A number of models have been made into the earlier works of the authors on single robots that revolve around evolutionary algorithms. In Kala, Shukla, and Tiwari (2010b) a hierarchical implementation of a customized evolutionary algorithm was proposed. A number of evolutionary operators were defined to result in

دریافت فوری ← **متن کامل مقاله**

ISIArticles
مرجع مقالات تخصصی ایران

- ✓ امکان دانلود نسخه تمام متن مقالات انگلیسی
- ✓ امکان دانلود نسخه ترجمه شده مقالات
- ✓ پذیرش سفارش ترجمه تخصصی
- ✓ امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
- ✓ امکان دانلود رایگان ۲ صفحه اول هر مقاله
- ✓ امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
- ✓ دانلود فوری مقاله پس از پرداخت آنلاین
- ✓ پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات