European Journal of Operational Research 242 (2015) 798–806

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

Search with evolutionary ruin and stochastic rebuild: A theoretic framework and a case study on exam timetabling

Jingpeng Li [a], Ruibin Bai [b,*], Yindong Shen [c], Rong Qu [d]

[a] Division of Computing Science & Mathematics, University of Stirling, Stirling FK9 4LA, UK
[b] School of Computer Science, The University of Nottingham Ningbo China, Ningbo 315100, China
[c] School of Automation, Huazhong University of Science & Technology, Wuhan 430074, China
[d] School of Computer Science, The University of Nottingham, Nottingham NG8 1BB, UK

ARTICLE INFO

Article history:
Received 29 December 2013
Accepted 2 November 2014
Available online 13 November 2014

Keywords:
Metaheuristics
Evolutionary algorithm
Stochastic process
Combinatorial optimisation
Exam timetabling

ABSTRACT

This paper presents a state transition based formal framework for a new search method, called Evolutionary Ruin and Stochastic Recreate, which tries to learn and adapt to the changing environments during the search process. It improves the performance of the original Ruin and Recreate principle by embedding an additional phase of *Evolutionary Ruin* to mimic the survival-of-the-fittest mechanism within single solutions. This method executes a cycle of *Solution Decomposition*, *Evolutionary Ruin*, *Stochastic Recreate* and *Solution Acceptance* until a certain stopping condition is met. The *Solution Decomposition* phase first uses some problem-specific knowledge to decompose a complete solution into its components and assigns a score to each component. The *Evolutionary Ruin* phase then employs two evolutionary operators (namely *Selection* and *Mutation*) to destroy a certain fraction of the solution, and the next *Stochastic Recreate* phase repairs the "broken" solution. Last, the *Solution Acceptance* phase selects a specific strategy to determine the probability of accepting the newly generated solution. Hence, optimisation is achieved by an iterative process of component evaluation, solution disruption and stochastic constructive repair. From the state transitions point of view, this paper presents a probabilistic model and implements a Markov chain analysis on some theoretical properties of the approach. Unlike the theoretical work on genetic algorithm and simulated annealing which are based on state transitions within the space of complete assignments, our model is based on state transitions within the space of partial assignments. The exam timetabling problems are used to test the performance in solving real-world hard problems.

1. Introduction

By definition, a solution to a combinatorial problem is always made of components which are elaborately interlocked together. Not only should each solution component be a strong candidate in its own right but also it has to fit well with other components in the surrounding environment or current setting. To deal with these components, Schrimpf, Schneider, Stamm-Wilbrand, and Dueck (2000) proposed a technique called Ruin and Recreate (R&R) principle for some classical problems (including the traveling salesman problem, vehicle routing and network optimisation), and claimed that it could be a general approach for various combinatorial optimisation problems. Since then, R&R has been successfully applied to many different types of discrete optimisation problems, such as quadratic assignment (Misevicius, 2003), paratransit scheduling (Häll & Peterson, 2013), the permutation flow shop problem and one-dimensional bin packing problem (Burke et al., 2009), etc.

The R&R method uses the concepts of simulated annealing (Kirkpatrick, Gelatt, & Vecchi, 1983) or threshold accepting (Dueck & Scheuer, 1990) with large moves instead of smaller ones. The method is sometime called very large neighbourhood search, and thus bears some resemblance to variable neighbourhood search (Hansen & Mladenović, 1999) which obtains better results by a perturbation of an existing solution and a following improvement procedure. High quality solutions are obtained by applying this type of treatment frequently. Hence, R&R method can be thought of as an iterative process of reconstructions and improvements applied to solutions. The advantage of the method over the well-known random multi-start method is that, instead of generating new solutions from scratch, a better idea is to reconstruct a portion of the current solution to make use of the information gained from the previous search.

For simple structured problems such as the traveling salesman problem, the need of using large moves is not obvious, because there is no/little feasibility issue, and small moves are usually sufficient

* Corresponding author. Tel.: +86 574 8818 0278.
E-mail addresses: jli@cs.stir.ac.uk (J. Li), Ruibin.Bai@nottingham.edu.cn (R. Bai), yindong@hust.edu.cn (Y. Shen), rxq@cs.nott.ac.uk (R. Qu).

http://dx.doi.org/10.1016/j.ejor.2014.11.002
0377-2217/

for the algorithms to generate near optimal solutions. Of course, the larger the size of moves, the better the results would be but at the cost of much higher computation time. For instance, Iterated Lin-Kernighan, one of the best approaches for the traveling salesman problem, does use large local moves. However, for complex problems such as exam timetabling problems, difficulties arise if we always use such small moves, because local moves do not give small changes in the objective function: if taking one move from a current solution to its neighbouring solution, the qualities of the resulting solutions might be significantly different due to the ruggedness of the landscapes in these problem areas. The situation may be alleviated by a better choice of solution representation and neighbourhood function, but this study is beyond the scope of this paper.

Solutions of complex problems usually come with a number of soft and/or hard constraints, which makes it difficult to get just feasible solutions. Neighbouring solutions of complex schedules, for instance, are mostly infeasible solutions. Walking in such a complex fitness landscape from one feasible solution to another feasible neighboured solution would be hard. The common method of avoiding the infeasibility problem for many classical algorithms in the literature is to impose artificial penalty functions, but this method would typically make the algorithms stuck in solutions which are nearly feasible but are not allowed at all.

Naturally, one will think in such as paradigm: Ruin and Recreate. Unlike a local search which implements search by doing perturbation on a small number (usually up to 3) of components, we ruin a large portion of the solution and try to rebuild the solution as best as we can, with the hope that the new solution is improved. The R&R approach is based on this idea. Of course, the R&R can also destroy a small portion of the solution (say 1–2 components), then under this circumstance it behaves the same as an ordinary local search algorithm. Hence, it is reasonable to believe that problems with many side conditions, or with complex objective functions, are more tractable using special-designed large moves.

Based on the R&R principal, in this paper we embed some evolutionary features into the decision process and present a more advanced technique called Evolutionary Ruin and Stochastic Recreate (ER&SR). Its general idea is to break a solution down into its components and assign a score to each component by an evaluation function which works under dynamic environments. The scores (or fitness values) determine the chances for the components to survive in the current solution.

The ER&SR applies two operators of *Selection* and *Mutation* as the ruining strategies, trying to mimic the survival-of-the-fittest mechanism happening on single individuals (or solutions). Each component in the solution has to continuously demonstrate its worthiness to remain in the current solution (or environment). Hence in each iteration, some components would be treated not worth keeping. The evolutionary strategy adopted may also remove some worthy components with fixed or variable low probabilities. The removed component is then reintroduced by using a specific algorithm. The addition of a new component is determined by a dynamic evaluation function, which computes how well the candidate component would fit in with others that already exist in the current solution. The above processes are repeated together with the remainder of the classical R&R. Hence, search is based on an iterative improvement process of components evolution, solution disruption and reconstructive process.

The proposed ER&SR algorithm comprises the following four phases: *Solution Decomposition*, *Evolutionary Ruin*, *Stochastic Recreate* and *Solution Acceptance*. It executes these phases in sequence on a single solution until a predefined stopping criterion is met. The first *Solution Decomposition* phase is based on the fact that solutions of the combinatorial optimisation problems all consist of components which are intricately woven together. Each component in a current solution may not only be a strong candidate in its own right, but also need to fit well with other components. Hence, the key problem in this phase is about what measurement to use to evaluate the fitness of individual components. To address this, we may employ an expert's domain knowledge to break a solution down into components and assign a score to each. The higher the score, the fitter the associated component is.

The second *Evolutionary Ruin* phase is based on the consideration that the incumbent solution must be changed not only locally but also over a macroscopic scale, depending on the solution composition defined by the proceeding *Solution Decomposition* phase. This phase employs two evolutionary operators of *Selection* and *Mutation* to destroy a certain portion of the entire solution. The *Selection* operator removes some components based on Darwin's survival of the fittest mechanism, while the *Mutation* operator further removes a small number of components at random. Hence, the destroyed part of the solution would sometimes be large enough such that the impact of the "bomb" that is thrown on the solution will be noticeable not only locally but in the whole system. On the other hand, the destroyed part would sometimes be small enough so that at least a main portion of the solution (i.e. a skeleton) remains to ease the rebuild for the next solution.

The third *Stochastic Recreate* phase follows to reintroduce the removed components by a somewhat stochastic method in order to have a better chance to jump out of the local optima. The fourth *Solution Acceptance* phase selects a specific strategy to determine the probability of accepting the newly generated solution.

This paper presents a probabilistic model based on the state transition process between the abovementioned phases, and implements a Markov chain analysis to derive some theoretical properties of the approach. The well-known exam timetabling problems are used to test the availability of the approach in solving real-world hard problems.

The remainder of the paper is structured as follows. Section 2 elaborates a formal framework of the ER&SR, from the state transition point of view. Section 3 introduces an implementation of the ER&SW for exam timetabling, and Section 4 presents the computational results of this approach. Section 5 contains some concluding remarks and possible future work.

2. The ER&SR algorithm

Following the general introduction of the ER&SR in Section 1, this section presents its formal framework in a similar way to that of the evolutionary squeaky wheel optimisation (Li, Parkes, & Burke, 2011b), and then presents a Markov chain model for a simplified version of the algorithm. The basic idea of the algorithm was initially and briefly presented in Li, Qu, and Shen (2012b). The session brings a deeper insight on the working mechanism of the algorithm.

2.1. A formal framework of the ER&SR algorithm

Four phases are performed in sequence at each iteration of the ER&SR algorithm. They are *Solution Decomposition*, *Evolutionary Ruin*, *Stochastic Recreate* and *Solution Acceptance* (see the flow chart in Fig. 1).

The first phase, *Solution Decomposition*, does not change the current state on its own, since it only collects information about the local fitness value of each component of the current state. The *Evolutionary Ruin* phase (comprising two operators of *Selection* and *Mutation*) changes the present state by removing a set of selected components from a current solution, based on the results of the preceding *Solution Decomposition* phase. Neither the *Selection* operator nor the *Mutation* operator can directly result in a destination state, that is, a state in which each of the componential variables is assigned a value. What is obtained after an *Evolutionary Ruin* is an intermediate state in which at least one component is removed. An intermediate state corresponds to an incomplete assignment, with a set of unassigned components.