The Development of a Multi-threaded Multi-objective Tabu Search Algorithm

Peter Dawson[1], Geoff Parks[1], Daniel Jaeggi[1], Arturo Molina-Cristobal[2], and P. John Clarkson[1]

[1] Engineering Design Centre, Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK
[2] Electrical Machines and Drives Group, Department of Electronic and Electrical Engineering, University of Sheffield, Mappin Street, Sheffield S1 3JD, UK

Abstract. The reliance of Tabu Search (TS) algorithms on a local search leads to a logical development of algorithms that use more than one search concurrently. In this paper we present a multi-threaded TS algorithm employing a number of threads that share information. We assess the performance of this algorithm compared to previous multi-objective TS algorithms, via the results obtained from applying the algorithms to a range of standard test functions. We also consider whether an optimal number of threads can be found, and what impact changing the number of threads used has on performance. We discover that, contrary to the popular belief that multi-threading is usually beneficial, performance only improves in a few special cases.

1 Introduction

Whilst there has been much research into other types of algorithms for multi-objective optimization [1], Tabu Search remains an option under-represented in the volume of research in this area. Following the recent development of a multi-objective TS algorithm, presented in [2], there is an opportunity to investigate variants of this algorithm. We are specifically interested in a multi-threading TS algorithm. The integral part played by a local search in the TS algorithm points towards potential benefit from using many local searches at once. These parallel searches are known as threads, and together they form a multi-threaded algorithm. Single-objective multi-threading strategies have been heavily researched. An overview is given in [3] (see particularly Type 3 parallelisation), and examples of specific implementations of multi-threaded single-objective TS are presented in [4,5]. Research in multi-objective meta-heuristics is currently following the pattern of studies previously undertaken for single-objective problems, and the fact that multi-threaded single-objective TS implementations exist provides us with more motivation to develop multi-objective variants of such algorithms.

2 Background

2.1 Multi-objective Optimization

The problem we seek to solve is that of multi-objective optimization. We wish to find a vector $\mathbf{x}$ that minimizes all the components of the vector $F(\mathbf{x}) =$

S. Obayashi et al. (Eds.): EMO 2007, LNCS 4403, pp. 242–256, 2007.

$(f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}), ...)$. The components of $\mathbf{x}$ are known as design variables and each $f_i(\mathbf{x})$ is an objective function. We usually seek values for design variables only in a predetermined range, and there may also be functional constraints on the variables.

Each set of vectors, those of design variables and those of objective functions, forms a vector space. We call these variable/decision/design space and objective space respectively. When we talk of searching a space we mean variable space, whilst when we consider the outcomes of our optimization algorithm we almost always refer to objective space.

Our aim is to minimize all our objective functions. However, we almost always work with situations where different points in variable space minimize different objective functions. We therefore must use the idea of Pareto dominance. Consider two points in variable space, $\mathbf{x}$ and $\mathbf{y}$, with objective functions $F(\mathbf{x})$ and $F(\mathbf{y})$. We say $\mathbf{x}$ *dominates* $\mathbf{y}$ if no component of $F(\mathbf{x})$ is greater than the corresponding component in $F(\mathbf{y})$ and at least one component is smaller. If neither $\mathbf{x}$ dominates $\mathbf{y}$ nor $\mathbf{y}$ dominates $\mathbf{x}$ the two points are said to be Pareto equivalent.

Now our aim is clearer – we wish to find a set of Pareto equivalent points not dominated by any point outside the set. This represents the trade-off surface between objective functions, and it is impossible to say that any one point is better than another within this set without weighting the objectives.

2.2 Tabu Search

The basic idea of Tabu Search is that of a local search looking at one point at a time before moving to the next, and maintaining three memories along the way. These are termed respectively the Short. Medium and Long Term Memories (STM, MTM, LTM). TS was developed for single-objective optimization problems and this is reflected in the original use of these memories. The STM is used to store the points that have been recently visited and then stops the search from revisiting these points, hence the method's name. The MTM stores optimal and near-optimal points found during search. The LTM stores information about all points visited. An important feature of TS is that an 'uphill' move will be made in preference to converging at a local minimum, as "a bad strategic choice can yield more information than a good random choice"[6].

The implementation of a multi-objective TS algorithm in this paper follows closely that presented by Jaeggi et al. in [2]. The functions of the three memories change, and a fourth – an Intensification Memory (IM) – is added. These changes are detailed in Sect. 3.1. The functions of the memories must change slightly to be compatible with a multi-threading algorithm which shares information, but all changes endeavour to remain consistent with the basic tenets of TS.

3 Multi-threaded Multi-objective TS Implementation

The algorithm presented in [2] has a single thread starting at a given point. The search then proceeds in an iterative manner; at each iteration a new point is selected via a variety of methods. The standard method is the Hooke and Jeeves