Benchmarking and Comparison of Software Project Human Resource Allocation Optimization Approaches

Sultan M Al Khatib
School of Computing Sciences
University of East Anglia
Norwich Research Park
Norwich NR4 7TJ
United Kingdom
s.al-khatib@uea.ac.uk

Joost Noppen
School of Computing Sciences
University of East Anglia
Norwich Research Park
Norwich NR4 7TJ
United Kingdom
j.noppen@uea.ac.uk

ABSTRACT

For the Staffing and Scheduling a Software Project (SSSP), one has to find an allocation of resources to tasks while considering parameters such skills and availability to identify the optimal delivery of the project. Many approaches have been proposed that solve SSSP tasks by representing them as optimization problems and applying optimization techniques and heuristics. However, these approaches tend to vary in the parameters they consider, such as skill and availability, as well as the optimization techniques, which means their accuracy, performance, and applicability can vastly differ, making it difficult to select the most suitable approach for the problem at hand. The fundamental reason for this lack of comparative material lies in the absence of a systematic evaluation method that uses a validation dataset to benchmark SSSP approaches. We introduce an evaluation process for SSSP approaches together with benchmark data to address this problem. In addition, we present the initial evaluation of five SSSP approaches. The results shows that SSSP approaches solving identical challenges can differ in their computational time, preciseness of results and that our approach is capable of quantifying these differences. In addition, the results highlight that focused approaches generally outperform more sophisticated approaches for identical SSSP problems.

Keywords

Human Resource Allocation; Software Project Management; Optimization Techniques in Software Engineering; Comparative Study; Performance Evaluation

1. INTRODUCTION

Software development is a mixture of complex activities and the creation of any non-trivial software system generally requires multiple resources with a mix of skills, expertise, and knowledge. The assignment of those resources in a software development department to projects and tasks within those projects is one of the most critical tasks for a project manager, with limited resources, dependent tasks, and available skillsets needing to be considered to achieve an optimal project delivery time. This problem of staffing and scheduling a software project (SSSP) in order to minimize the project completion time has been attracting researchers since the end of last century [1-4] and different optimization techniques have been used to address it in various incarnations [3, 5, 6]. These approaches typically consider specific attributes when optimizing the resource allocation such as task length, resource availability or skills, and the traversal of the optimization space is typically performed by using exact, heuristic, and meta-heuristic techniques in order to deal the NP-Complete nature of the allocation problem [3]. Project managers typically can select an automated SSSP approach to support their allocation process based on the project and resource properties they wish to consider. However, approaches can have different performance characteristics such as the accuracy of the allocation results or computational time required, characteristics that are critical for successful SSSP but very hard to determine without a systematic manner. Limited number of studies in this context [3, 4] were published that compare SSSP approaches but neither of these studies performs an empirical evaluation of SSSP approaches using a unified basis and data set.

This article proposes to address that gap by introducing a benchmark and using it to evaluate the performance of a set of SSSP approaches against well-defined performance measures. Specifically, we aim to provide a validation dataset that has both resources and detailed project information for a range of SSSP challenges. In addition, we aim to compare the SSSP approaches using a uniform and expandable set of performance measures that can compare SSSP approaches in various categories and supporting a range of optimization criteria.

In addition to the benchmark and initial results of the comparison analysis in this article, we also outline our research agenda. To further the accuracy and relevance of the performance evaluation we aim to perform a comparison of computational approaches and current industry standards. This will be complemented with the implementation and evaluation of additional SSSP approaches to form a complete and comprehensive overview of SSSP approaches as well as the means to perform systematic comparisons between them. Note that this should not be confused with the comparison of the heuristic algorithms. The comparison adopted in this paper considers the approaches that propose a model for allocating the developers in software projects with modification on the algorithms they use.

The remainder of this paper is organized into five sections. Section 2 describes the studies carried out in comparing SSSP approaches that are related to the work presented in this paper. Section 3 detailed the workflow of procedures, dataset, criteria proposed to evaluate and compare the SSSP approaches, future plan of carrying out the rest of study work, and the threats and weaknesses that could affect the validity of this study. In section 4, the approaches adopted in this study are described and the results of the experiments and comparison between the SSSP approaches are shown. Section 5 discusses the main findings and concludes the paper.

2. RELATED WORK

When considering previous work performed in the area of evaluating SSSP approaches, only two studies have been published that compare and evaluate the optimization approaches of SSSP.

Both comparison studies were based on evaluating the approaches according to the description provided within the texts. These studies have compared the approaches by a comprehensive survey [3] or systematic literature review [4] by extracting the text that describing the problem and solution of the approaches. Thus, these studies are more formally systematic literature review with comprehensive survey of wide software project management approaches.

The first study by Pixoto et al [4] evaluates the solution provided by SSSP approaches regarding their applicability in real-world software development projects. Criteria used by Pixoto et al to evaluate the description of solutions are usefulness, work compatibility, and ease of use attributes. 52 approaches were considered by this study. The comparison shows that few approaches among them all are satisfying the criteria adopted and capable for the illustrated aspects by this study as the one in [7]. Skills and productivity of resources found are the least aspects considered by the approaches used by Pixoto et al [4]. In addition, time and cost of software projects are the goals adopted by overwhelming majority of SSSP approaches. It is also noticeable in this study that only 8% of the approaches compared found they have used experiments to validate their solution. The overall conclusion by this study is that more research is needed to bridge the gap between the current practices of software firms and the proposed solutions. As this study provides essential aspects and differences between the SSSP approaches, the adoption model of criteria and aspects used are based on theoretical models. Criteria and aspects however have to be validated by the industry before they can make their claims about the usefulness of the approaches used in their study.

The second study presented in Ferucci et al [3] provides a comprehensive survey of the approaches use optimization techniques to solve software project management problems. Their observations and findings highlight the categories of the optimization approaches, the important attributes that these approaches adopted, and the approaches that match their criteria and seen useful to be adopted. The approaches used by this study are categorized into minimizing project time, risk-based, overtime planning, and effort estimation. This study has also identified the future trends and promising areas of resource allocation optimization. The areas found require more attention by researchers as future trends are interactive optimization, dynamic adaptive optimization, multi-objective optimization, co-evolution, software project benchmarking, confident estimates, and decision support tools. While this study is a comprehensive survey, it can be seen as a general study that reports the different types of problems adopted by approaches deal with software project management with no consideration of further classification or either cross functionality between the approaches and how each has opened a new knowledge.

The results presented in these studies are a valuable insight into the relation between various SSSP approaches, however neither study performs a systematic comparison between the SSSP approaches considered based on their implementation and a reference dataset. This is due to the fact that a benchmark dataset currently is not available in this research area. While two repositories exist for the use of software engineering research, which are ISBSG and Tera-PROMISE, none of these includes a valid dataset containing human resource models and detailed project information usable for SSSP based research [3]. Accordingly, there is an urgency in this particular area for a data that represent a real software project to benchmark the SSSP approaches [3]. As a result, comparing and benchmarking SSSP approaches based on their behaviour and performance has not been carried out even when it has been identified as highly important by the community [3].

3. A SYSTEMATIC APPROACH FOR COMPARING SSSP APPROACHES

3.1 Overview of the Proposed Approach

Our proposed approach for performing a systematic and reproducible performance comparison of SSSP approaches consists of a systematic sequence of steps to be followed combined with an evaluation dataset and a suite of evaluation criteria on which the SSSP approaches can be compared. The proposed workflow for evaluating a set of SSSP approaches consists of the following steps:

1. Select a set of candidate SSSP approaches that are capable of solving a resource allocation problem and belong to the same class – see section 3.2 -.
2. Select the suitable dataset from the benchmark dataset that belong to the same class of the approaches selected containing the desired resource and project properties (e.g. skills, task dependencies, etc.)
3. Run each approach for the configured dataset for a substantial number of times, (e.g 100 times).
4. Record for each run the result of estimated project time, and the computation time of that run (see below).
5. Compile the results and measure their performance using the benchmark metric suite (see below).
6. Rank the candidate SSSP approaches based on their score in the overall scoring model (see below).

These steps are depicted in Figure 1. As can be seen in the Figure, after identifying the approaches, the classes that they belong to, and selecting the suitable benchmark dataset, the datasets located on the left down of the figure is fed into each approach. As most approaches perform heuristic optimization using a probabilistic optimizer, step 3 suggests to perform multiple runs for each of those approaches so that their computation time and accuracy can be averaged, as well as their mean and standard deviation can be determined. The choice for these metrics is motivated by the fact that they are seen as the most useful way to represent effectiveness and performance among the approaches [8].

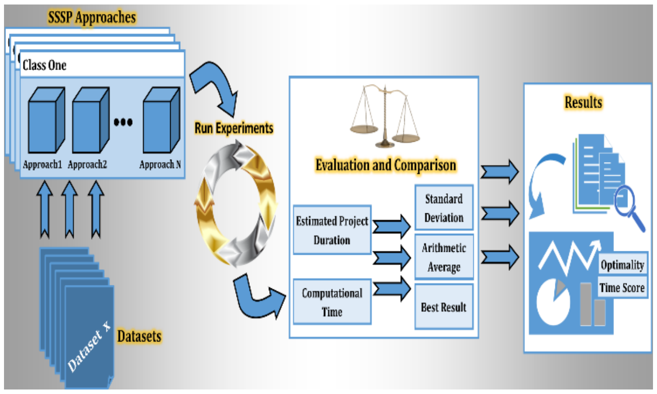

Figure 1: Proposed Approach

3.2 Benchmark Dataset

The first artefact we introduce to perform a systematic evaluation of SSSP approaches is a flexible and configurable benchmark dataset. The dataset is a small real world data from a Jordanian software company and holds information regarding both software project and human resources used to develop that software. This data includes information about eight components of the software projects, and twelve human resources were available to that project assigned to complete it. The project represented in the dataset has an estimated time using COCOMO. The time estimated with those resources available was 75.16 days, with an estimated Man-Day equals to 964. The dataset is composed of five sets the first four correspond to the classification made to the SSSP approaches. The first four sets describe resource allocation problems of increasing complexity and parameters. The final set describes a resource allocation problem of a larger size that is intended to analyse the scalability of the approaches in class 1. In addition, for each one of these classes the optimal solution (referred to as *min* value) as well as the worst-case solution values (referred to as *max* value). The dataset used in this article can be found on http://seg.cmp.uea.ac.uk/projects/resource-optimisation/files/dataset.zip.

When benchmarking SSSP approaches, it is critical to note that proposed approaches generally solve different variations of the resource allocation problem, taking into account different parameters, such as worker skills, or tasks dependencies. To evaluate the relative performance of SSSP approaches they need to be applied to the same problem with the exact same inputs, which is why we propose to group SSSP approaches into classes according to the inputs and constraints required by each. The inputs required for resource allocation can be the estimated effort of project tasks, task dependencies, skills, and/or resource productivity. Each one of these inputs represented in the dataset by numbers except the skills. Skills required for developing each task or offered by a resource are representing languages and technologies, and represented in the dataset using the name of this language or technology such as java, or UML. Estimated effort of each task is represented by person-day. Each task in the dataset moreover has the value of dependency attribute represented as the task number that the task is depends on. Productivity of a resource is represented by the same metric used by [7]. A resource can be productive as a normal person, which is equal to 1, or twice the normal person represented by 2. According to these inputs the proposed classes are:

- **Class One**. This class contains the approaches that require inputs only of estimated effort of project tasks and the number and productivity of human resources.
- **Class Two**. This class contains the approaches that require inputs of estimated effort of project tasks, dependencies between these tasks, and number and productivity of human resources
- **Class Three**. This class contains the approaches that require inputs of estimated effort of project tasks, skills required for each tasks, and number, skills, and productivity of human resources
- **Class Four**. This class contains the approaches that require inputs of estimated effort of project tasks, dependencies between these tasks, skills required for each tasks, and the number, skills, and productivity of human resources.

Note that some SSSP approaches can possibly be part of multiple classes as they are able to determine the optimal allocation of resources for simple as well as complex SSSP problems. The performance for such approaches can be compared to other approaches in both classes with respect to solving identical problems. The benchmark data follows this classification as it defines optimization challenges within these five distinct classes to facilitate the uniform comparison of SSSP approaches

3.3 Comparison Metrics and Overall Scoring Model

The performance of a SSSP approach is usually measured in terms of optimality, i.e. how close the approach gets to the true optimal solution [9]. However, this metric only provides a partial view. For example, many probabilistic optimizers, such as genetic algorithms, vary in the quality of solution they provide due to a randomised starting point and the computation time expended to them. Accordingly, both of resulted values from the approach for the objective function -which in this study is the estimated project time- and the computational time expended to produce the results are the main metrics of this comparison. In addition to the performance measures of optimal solution and computation time, behaviour of the approaches have to be recorded too. While each approach uses a modified version of optimization technique, it is important to capture stability and preciseness of the approach over multiple runs. The importance of having a multiple runs is due to the probabilistic nature of meta-heuristic algorithm search. This can be depicted by the standard deviation of multiple runs of both estimated project time and computational time. To get a more complete insight into the performance of SSSP approaches we propose to use the following metrics:

1. **Estimated Project Time (EPT)**. The first proposed metric is the estimated project time, i.e. the identified optimal result by an approach for each run.
2. **Computational Time (CT)**. Computation time is the time consumed by the system to perform the approach from the point of feeding the data to the time of identifying the (heuristically) optimal result.
3. **Standard Deviation (STDEV)**. This metric is the standard deviation among the collected EPT values. This metric is a useful indicator of whether an approach is robust and precise. As the standard deviation will quantify outcomes produced are closely grouped or not.
4. **Arithmetic average (Mean)**. The mean of values resulting for an SSSP approach over multiple runs.
5. **Minimal EPT**. The least possible value for estimated project time among the collected values over multiple runs.

Note that metrics such as STDEVB and mean require the performance of the approach to be determined over multiple runs so that the average behaviour can be established and compared.

In addition to this suite of metrics, we propose the use of an overall scoring model for easy comparison of SSSP approaches, consisting of two formulas. The first formula captures the accuracy of a SSSP approach using the following equation:

$$\textit{Optimality of solution} = [1-[(V-min)/(max-min)]] \times 100$$

This formula depicts how close the value calculated by a SSSP approach (V) is to the known optimal solution (*min*).This value is normalised using the known worst-case solution (*max*). Both the *min* and *max* values are included in the dataset for a given SSSP problem. In addition, a model for scoring the computational time performance of an approach is depicted by the following equation.

$$\textit{CTime Score} = [\, Vct \,/\, Max\,(Class)]$$

In this formula Vct is the computation expended by approach V to solve the SSSP problem under consideration of Max(Class) which is the maximum computation required for all known SSSP approaches capable of solving this problem.

3.4 Research Agenda for Comparison Benchmark of SSSP approaches

The work described in this paper is a first step towards a systematic mechanism for evaluating SSSP approaches with respect to their performance and accuracy. The research plan from this point focuses on extending the SSSP benchmark method and evaluating its usability and applicability in an industrial setting. To this purpose, the research plan is divided into four parts:

- The first part is the refinement of the benchmark dataset to include more projects and resource data as well as a refined configuration mechanism that allows for easy configuration.
- Second we aim to extend the set of implemented and evaluation SSSP approaches to provide a comprehensive set of data points that researchers can use to compare their own approaches to.
- Thirdly, we aim to examine a mechanism that allows us to easily bridge the gap between SSSP approaches so users of the benchmark can more easily evaluate a range of SSSP approaches against a set problem with specific parameters.
- Finally, upon establishing a reasonable and balanced SSSP benchmarking process we will evaluate its suitability and relevance by means of empirical evaluation with industrial partners. The results of the experienced project managers in allocating resources to projects will be compared to SSSP approaches and their benchmarking results for this purpose.

3.5 Threats to Validity and Challenges in comparing SSSP approaches

One of the main threats to validity in this study is that the data collected represents a single use of allocation attributes of one software firm, which can have an implication regarding the validity of the comparison with the different styles adopted in the industry regarding the allocation, constraints, and the development method within these firms. However as the dataset used to compare the approaches is a real-world data, it represent a small project which might not be the common scenario in software firms and the capabilities offered by various types of SSSP approaches are not covered such as dealing with a massive software project. Moreover, extending it to cover the capabilities of SSSP approaches while at the same time remaining representative can be very challenging. Thus, we aim to ensure the relevance of the data, and the approaches by expanding the experiments with our industrial partners. A further threat to the relevance of our evaluation results is the limited detail provided by publications describing SSSP approaches. In many cases, vital elements of the approach are not described sufficiently and no reference implementation of the approach is provided for evaluation. We have addressed this threat in our approach by excluding approaches with incomplete descriptions that prevented us to implement it. Where possible we have liaised with the authors of the approach to clarify ambiguities and complement the publication.

4. BENCHMARK APPLICATION TO EXISTING SET OF SSSP APPROACHES

4.1 Overview

To assess the accuracy and suitability for our proposed approach and benchmark we have performed a preliminary study of five SSSP approaches in two different classes. The approaches focus on optimizing the software project time using meta-heuristic techniques such as Genetic Algorithm (GA) and Simulated Annealing (SA) while taking into account various parameters such as task dependency to find the optimal or near optimal project time. The reason for selecting these approaches in this comparison is based on the studies presented in [3, 4]. These approaches are presented in Table1 according to the class they belong to. The approaches have been classified according to the SSSP classes introduced in Section 3.2. The optimization techniques used by the approaches are Genetic Algorithms (GA) by [5, 6, 10, 11], and a modified version of Simulated Annealing (SA) called Accelerated SA by [7]. Both techniques are belong to the same search algorithm class called meta-heuristic.

Table 1: Approaches Classification

Class	One	Two	Three	Four	Five
Approach					
[10]	**X**				
[11]	**X**				
[7]	**X**		**X**		**X**
[5]		**X**			
[6]		**X**		**X**	

Work has been accomplished to classify the approaches described earlier according to the classes they can use. This table shows the applicability of dataset classes too for each approach described earlier.

4.2 Results

The results were obtained using the Matlab R2013a supported by Matlab Global Optimization Toolbox using Intel Core 2 quad 2.66 Ghz CPU. Each approach was executed 100 times to allow determination of mean and deviation values. The comparisons performed were between Di Penta et al [10], Di Penta et al [11] and Kang et al for the Class 1 benchmark data, and between Chan et al and Alba et al for the Class 2 dataset.

4.2.1 Results of the Class One Dataset Evaluation

The first results we present are for the Class 1 approaches [10],[11], and Kang et al [7]. The dataset used is the Class One dataset, which only considers tasks, resources and availability, and has an optimal solution of 80.33 for its project schedule. Figure 2 shows how each iteration for each approach resulted an EPT in term of days where the lowest value amongst the approaches is the one obtained by DiPenta et al [10]. Moreover, we can see that the approach in both DiPenta [10] and [11] were quite close to the estimate of COCOMO presented in Section 3.3.

The results obtained for Kang et al approach on the other hand is overestimating project time when compared to any one of the DePinta el al approaches. This is due to the allocation method adopted by Kang et al approach as it assigns single resources to tasks with least estimated effort, where those that have the biggest effort required are each assigned to two resources which results in a less accurate approximation. The numeric results for accuracy are given in Table 2. It is interesting to observe that DiPenta et al [10] is the most accurate and it has managed to identify the actual optimal solution (80.33) for the dataset task. DiPenta et al [11] has come close to finding the optimal solution but Kang et al struggled to come close. A graphical representation of this data as well as the behaviour over multiple runs can be found in Figure 2.

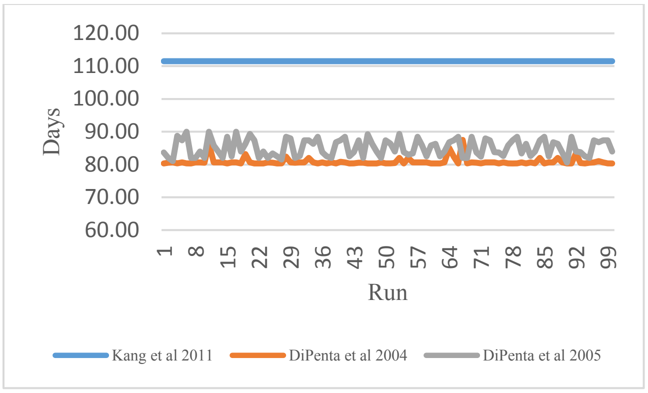

Figure 2: Accuracy performance over a 100 runs for Class One

When we examine the computation time results in Table 2. It can be seen that DiPenta et al [11] is the least time consuming among the approaches whereas Kang et al requires slightly more time. DiPenta et al [10] clearly requires the most time to identify an optimal solution.

Table 2: Performance results of Class One

Approach	Computation Time		Accuracy of Solution		
	Mean	STDEV	Mean	STDEV	Minimal EPT
[7]	127.90	2.82	111.5	0	111.5
[10]	285.91	2.57	80.83	1.139	80.33
[11]	109.65	0.19	85.13	2.61	80.6

An interesting observation as well is that while DiPenta et al [11] is not only faster, its standard deviation also is significantly lower than the two other approaches, which means the optimization behaves more uniformly in repeated experiments. This is a quality attribute that can become important when the problem size is scaled up, as a small variation in computation time can make solving a particular problem infeasible.

4.2.2 Results of the Class Two Dataset Evaluation

For the Class 2 approaches [5, 6] their performance was evaluated using the Class 2 dataset, where constraints are imposed on project schedule corresponding to dependencies between tasks. This dataset has an optimal solution of 81.95 days for the project schedule. When examining the results in Table 3. It can be seen that the approach of Chang et al is capable of identifying the optimal solution where the approach by Alba et al is not, however the approach of Alba et al gives a more reliable and reproducible results for a single run, as illustrated by the standard deviation value. This becomes even more clear when examining Figure 3 where Chang et al clearly fluctuates per run where the results of Alba et al is more tightly grouped together.

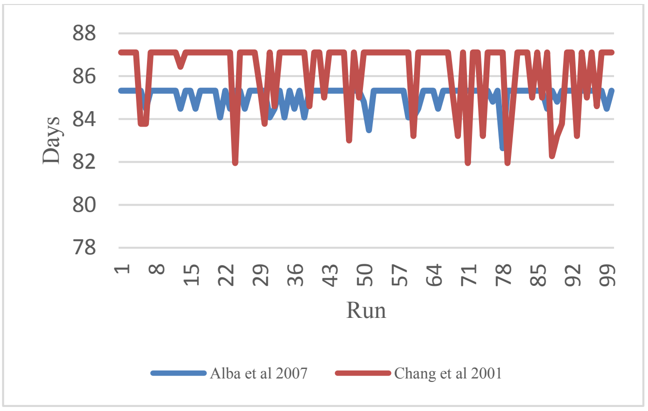

Figure 3: Accuracy performance over a 100 runs for Class Two

An interesting picture surfaces when we examine the computation time required by both approaches, as depicted in Table 3. It can be seen that while Chang et al fluctuates in the accuracy of the answer returned per run, on average it completes significantly faster than Alba et al. In this case, it is clear that while both approaches apply similar techniques Chang et al have sacrificed part of their accuracy for improved computation time performance.

Table 3: Performance results of Class Two

Approach	Computation Time		Accuracy of Solution		
	Mean	STDEV	Mean	STDEV	Minimal EPT
[5]	41.88	0.17	86.29	1.52	81.95
[6]	134.99	1.91	85.1	0.49	82.64

4.3 Ranking SSSP Approaches Comparison Using the Scoring Model

As the final step of our preliminary evaluation, we rank the evaluated SSSP approaches using our proposed scoring model. By combining the results of the approaches using the computation time and estimated project time and the formulas presented in Section3.3 we can compile the results in Table 4.

Table 4: Ranking results for the approaches

Class	Approach	Optimality of Result	CT Score
Class One	[7]	96.5%	0.45
	[10]	99.9%	1
	[11]	99.46%	0.3835
Class Two	[5]	99.37%	0.312
	[6]	99.54	1

This table gives an aggregated overview of the evaluation results using our dataset and metric suite. It can be seen for the Class 1 approaches that both approaches proposed by DiPenta et al are very close in accuracy but differ in computation time, with Kang et al representing a middle ground. For Class 2 a clearer winner can be identified with Chang et al offering similar accuracy to Alba et al but requiring far less time. We imagine that this aggregated scoring model will aid practitioners in comparing SSSP approaches and as such, it is one of the important deliverables of our research. Note however that in this scoring model at the moment the added value of standard deviation for both accuracy and computation is lost. In future work, we aim to include these explicitly in the scoring model to give a more complete picture.

5. CONCLUSIONS

In this article, we have identified that many different optimization approaches exist for staffing and scheduling a software projects (SSSP), but due to differences in the problem parameters they can consider as well as the optimization techniques they use their performance and applicability can be hard to assess and compare. To address this issue we have introduced a systematic comparison method for SSSP approaches together with a set of comparison metrics and an overall scoring model that can be used to rank their performance. This comparison method is combined with a benchmark dataset and reference values that identifies and supports four different classes of SSSP approaches based on their capabilities and limitations. We have applied our method and benchmark data to a set of five SSSP approaches and from these early results the applicability and accuracy of our method became clear. Our method highlighted that focussed approaches that aim to solve a well-defined SSSP problem are more likely to identify an accurate solution within a reasonable amount of time rather than approaches that can potentially consider a wider range of parameters and inputs.

Our future work and the expected contribution of my dissertation lies first in the creation of a more comprehensive method and reference dataset for comparing SSSP approaches but also in evaluating this with industry experts who are expected to apply the method in practice. To achieve this we are planning further experiments and evaluation with the intention to expand the dataset and add support for the remaining SSSP classes. In addition, we aim to expand the range of SSSP problems per class in both complexity and size to aid in the evaluation of scalability. Finally, we aim to perform an empirical experiment where we ask industry experts to apply and evaluate various SSSP approaches and compare the results to the evaluation results of our method to establish the relevance and accuracy of the method in real-world application scenarios. Our eventual goal for this work is to serve as an accurate and flexible reference mechanism for both academics and practitioners for determining the performance and accuracy of SSSP approaches.

6. REFERENCES

1. Tsai, H.-T., H. Moskowitz, and L.-H. Lee, *Human resource selection for software development projects using Taguchi's parameter design.* European Journal of Operational Research, 2003. **151**(1): p. 167-180.
2. Di Penta, M., M. Harman, and G. Antoniol, *The use of search-based optimization techniques to schedule and staff software projects: an approach and an empirical study.* Software: Practice and Experience, 2011. **41**(5): p. 495-519.
3. Ferrucci, F., M. Harman, and F. Sarro, *Search-Based Software Project Management*, in *Software Project Management in a Changing World*, G. Ruhe and C. Wohlin, Editors. 2014, Springer Berlin Heidelberg. p. 373-399.
4. Peixoto, D.C., G.R. Mateus, and R.F. Resende. *The Issues of Solving Staffing and Scheduling Problems in Software Development Projects*. in *Computer Software and Applications Conference (COMPSAC), 2014 IEEE 38th Annual*. 2014. IEEE.
5. Chang, C.K., M.J. Christensen, and T. Zhang, *Genetic algorithms for project management.* Annals of Software Engineering, 2001. **11**(1): p. 107-139.
6. Alba, E. and J.F. Chicano, *Software project management with GAs.* Information Sciences, 2007. **177**(11): p. 2380-2401.
7. Kang, D., J. Jung, and D.H. Bae, *Constraint-based human resource allocation in software projects.* Software: Practice and Experience, 2011. **41**(5): p. 551-577.
8. Kwok, Y.-K. and I. Ahmad, *Benchmarking and comparison of the task graph scheduling algorithms.* Journal of Parallel and Distributed Computing, 1999. **59**(3): p. 381-422.
9. Kwok, Y.-K. and I. Ahmad, *Static scheduling algorithms for allocating directed task graphs to multiprocessors.* ACM Computing Surveys (CSUR), 1999. **31**(4): p. 406-471.
10. Antoniol, G., M. Di Penta, and M. Harman. *A robust search-based approach to project management in the presence of abandonment, rework, error and uncertainty*. in *Software Metrics, 2004. Proceedings. 10th International Symposium on*. 2004. IEEE.
11. Antoniol, G., M. Di Penta, and M. Harman. *Search-based techniques applied to optimization of project planning for a massive maintenance project*. in *Software Maintenance, 2005. ICSM'05. Proceedings of the 21st IEEE International Conference on*. 2005. IEEE.