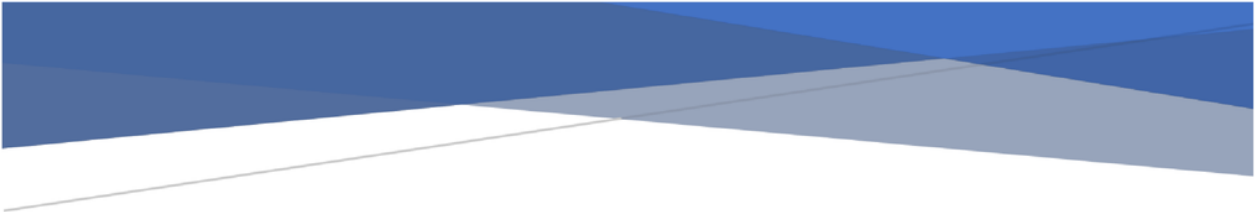


Project Proposal

by Nicholas Hinder

FILE	2927_NICHOLAS_HINDER_PROJECT_PROPOSAL_50131_547028611.DO CX (117.35K)		
TIME SUBMITTED	03-MAY-2019 04:23PM (UTC+0100)	WORD COUNT	2219
SUBMISSION ID	106266119	CHARACTER COUNT	12112



AUTOMATIC PARALLELISM OF EMBARRASSINGLY PARALLEL PROBLEMS

Project Proposal

Nick Hinder
P148207

Contents

Introduction	2
Background	2
Project Aims and Expectations	2
Research & Development	3
Philosophy.....	3
Strategy	3
Risks	4
Evaluating Data	4
Resources & Planning	5
Resources	5
Timescales & Coordination	5
Bibliography	6

Introduction

Background

It has been said that the time of the “free lunch” is over in regards to processor performance; this is due to the ever increasing demands of solutions out-pacing the now slowing improvements to performance in new processors. Sutter (1) proposed this era was coming to an end in his 2005 article on the subject; he states that the “free lunch” will continue a while longer but that in order to get the same gains as previously there would be development required and an increased complexity.

In the decade since this prediction, Sutter’s words have been proven to be correct in that whilst technology continues to advance; in order to obtain the desirable performance improvements increasing amounts of development is required as demands continue to grow. As a result of this programmers have, in recent years, been turning more and more to parallel computing as a solution to the issue.

However parallel solutions are, as previously mentioned, more complex than serial solutions and as such sometimes go unidentified or can be difficult to justify for some projects. The difficulty in introducing parallel solutions is identifying code that not only can be made parallel, but would also benefit from being made parallel.

Parallel programming introduces problems such as synchronisation issues and resource management difficulties. It is important to ensure that when parallel programming is introduced to a solution, that the solution does not suffer due to common parallel programming issues such as race conditions, deadlocks or parallel access problems such as the ABA problem (whereby two parallel operations modify a shared resource but due to the parallel nature do so without knowledge of the others changes).

Parallel programming itself has limitations in how much of a performance increase is available, primary of which is Amdahl’s law:

$$S_{max} = \frac{1}{(1 - p) + \frac{p}{s}}$$

Which demonstrates that, whilst more parallel threads can improve the speed increase gained from parallelisation, the max speed increase is primarily limited by the proportion of the code that cannot be parallelised.

Project Aims and Expectations

The project aims to create a software solution that will facilitate automatic parallelisation of embarrassingly parallel problems. The solution must:

- Not lead to erroneous behaviour in the sample code
- Lead to performance benefits when used with sufficiently large, parallelisable samples
 - This performance benefit will be measured in terms of execution time and should demonstrate a benefit that is at least 80% of the expected improvement for each sample in comparison to the expected target yielded by Amdahl’s law.
- Be usable with different code samples rather than tailored to individual samples

The solution should:

- Not cause performance deficits regardless of sample size
- Allow for future development to accommodate more complex parallel solutions

- Create dynamic code that runs at least 90% as swiftly in comparison to a manually parallelised solution

Research & Development

Philosophy

This project will be implemented with a positivist philosophy due to the inherent nature of its aims. The project proposes to develop a software solution to automatically parallelise code, as such it is possible to measure the run times of samples before and after the software solutions application. The resultant comparative data will allow for an appropriate deduction of the efficacy of the solution in its aims of speeding up code for embarrassingly parallel programs automatically.

Strategy

This project relies solely on the development of software that should meet the aim of speeding up code samples by automatic parallelisation. As such it is imperative to ensure that enough data is gathered to demonstrate that the results are accurate and reliable.

The development of a larger number of different test cases will be important in testing the solutions robustness and adaptability, whilst also providing more comparative data in order to demonstrate the resultant speed-up of the code provided by the automatic parallelisation.

The data gathered from the tests can be will form the basis for quantitative data analysis; in a bid to reduce observer bias, this data will be included in the project both in its raw form and post evaluation to support conclusions drawn from the data.

The solution will be developed in C# using the .Net framework. C# and the .Net framework are in great usage throughout the software development industry and as such; through targeting the such a solution with this project, the overall impact the solution could result in is greater due to the greater target audience.

C# and the .Net framework lend a number of features that are not present in other technologies, foremost examples of which are garbage collection (the automatic destruction of variables and data constructs that are no longer referenced) and Language Integrated Query (LINQ). These features have led to C# becoming an industry standard language due to the rapid, effective and safe development of code these tools permit. The .Net framework also encapsulates the Task Parallel Library (TPL) which facilitates parallelism in the .Net framework through task based programming and supports parallel LINQ (PLINQ) so as to allow parallelism when dealing with data objects.

In order to interpret and parallelise code samples reflection will be employed; according to the reflection article in Microsoft's documentation (2) *"Reflection provides objects (of type Type) that describe assemblies, modules and types."*, thereby showing that reflection can be used to read a given code. The article continues to state *"You can use reflection to dynamically create an instance of a type, bind the type to an existing object, or get the type from an existing object and invoke its methods or access its fields and properties."*, demonstrating that it will also be possible to write a dynamic method and invoke even invoke said method.

The detection of parallelisable code is a complex subject due to the many different patterns of parallelisable code that are available. The work by Astorga et al (3) on pattern analysis in a c++

application focused on the map, farm and pipeline parallel patterns; this work demonstrates that such a solution is possible with other technologies and as such provides a good example for this.

The primary patterns that will be matched by this project will be these three patterns with the aim of future work to expand the project to accommodate other patterns in addition to these.

Risks

There are several risks to the successful completion of this project, primarily of which are:

Timescales and coordination: these will always pose a risk in any project, however as a part time student this is a significant risk that must be controlled. This will be discussed further in the Resources & Planning section.

Complexity: this project is based on the automatic parallelisation of code, this entails not only a firm understanding and proficiency with writing parallel code but also with using reflection and dynamic code; all of which are advanced software techniques. The complexity of the project itself presents a risk in that this complexity could contribute towards the project not being completed in time.

Data loss: as with all projects, the potential for loss of data is present and is a major risk to the completion of the project. The project being a software solution is prone to this as any results gathered. In order to control this risk appropriate backup solutions will be utilised; the results and other work will be backed up to cloud services such as Google Drive whilst the software will utilise a remote source control repository using services such as GitHub.

C# & .Net Framework overheads: these technologies, whilst widely employed as a result of their ease of use and features, have a higher resource requirement than other technologies such as C++. There is the risk that performance improvements may be lessened due to the inherent overheads, or that there may be some impact from inherent behaviours of the technologies such as the non deterministic behaviour of C# garbage collection.

Evaluating Data

In order to properly evaluate the data, the tests against the varying code samples must be carried out and recorded accurately.

Each code sample will be run 10 times in order to determine an average baseline runtime for the samples.

Each code sample will then be run 10 times with the automatic parallelisation solution in effect in order to determine the resultant runtime for the samples after having parallelisation applied.

Simple comparisons between the baseline and resultant readings can be used to determine the average speed changes. Whilst the improvements should vary from sample to sample, it should be possible to decipher a positive trend and it should also be possible to average the increases to find one overall figure for the increases in regards to all samples.

The number of samples serve the primary purpose to reduce the impact of rogue data and increase the reliability of the data and by extension the conclusions drawn from it.

Resources & Planning

Resources

In order to undertake this project I will require an appropriate Integrated Development Environment (IDE) for developing the solution; the proposed IDE for this project is Microsoft Visual Studio. Microsoft Visual Studio Community will suffice for the development of this project, however a single professional license is also available for \$499 or \$45 a month. The decision to use Visual Studio Community rather than Visual Studio Professional is primarily a cost benefit, but one that also will entail less administration due to the ease of use regarding Visual Studio Community licenses.

Visual Studio Community will suffice for the projects aims, the Microsoft Visual Studio Licensing document (4) states that *"An unlimited number of users within an organization can use Visual Studio Community for the following scenarios: in a classroom learning environment, for academic research, or for contributing to open source projects"*. This clearly shows that not only the student, but also the university, are permitted usage of the IDE for this project.

Timescales & Coordination

Due to work obligations, the timescale of this project is of even more importance as work must be properly planned in order to be achievable around a full time work schedule.

Upon commencing the project the first stage of planning will be to implement an achievable, realistic Gantt chart to follow throughout the projects length.

There will be a period of time spent researching topics such as reflection and parallel patterns that will be required knowledge before proceeding to the main bulk of the project.

The majority of the time of the project will be spent in development and testing iterations. First parallel solutions can be created for the test cases and tested. Following this work on a solution for automatic generation of such parallel code will be carried out; this is where the majority of the work will take place and as such will be where the majority of time will be allocated.

Some time will be allocated towards the end of the project for proper testing and analysis of results alongside drafting the report and detailing the results and conclusions.

Due to the part time nature, the commencement date for the project is as yet unknown and as such it is not possible to perform such estimation of timescales and planning. Instead this would be carried out immediately upon confirmation of the projects commencement.

Bibliography

1. *The Free Lunch Is Over. A Fundamental Turn Toward Concurrency in Software.* **Sutter, Herb.** 30, s.l. : Dr. Dobbs's Journal, 2005, Vol. 3.
2. **Microsoft.** Reflection (C#). *Microsoft.* [Online] 20 July 2015. <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/reflection>.
3. *Finding parallel patterns through static analysis in C++ applications.* **Astorga, David del Rio, et al.** 6, s.l. : The International Journal of High Performance Computing Applications , 2016, Vol. 32.
4. **Microsoft Corporation.** Microsoft Visual Studio Licensing. *VisualStudio.Microsoft.* [Online] <https://visualstudio.microsoft.com/wp-content/uploads/2017/11/Visual-Studio-2018-Licensing-Whitepaper-November-2017.pdf>.
5. **Raman, Arun, et al.** *Speculative Parallelization Using Software Multi-threaded Transactions.* s.l. : Departments of Electrical Engineering and Computer Science, Princeton University, 2010.
6. **Aldinucci, M., et al.** *Design patterns percolating to parallel programming framework implementation.* s.l. : Springer Science, 2013.
7. **Aldinucci, Marco, et al.** *Pool Evolution: A Parallel Pattern for Evolutionary and Symbolic Computing.* s.l. : Springer Science, 2015.
8. **Fonseca, Alcides, et al.** *Automatic Parallelization: Executing Sequential Programs on a Task-Based Parallel Runtime.* s.l. : Springer Science, 2016.
9. **Otoni, Guilherme, et al.** *Automatic Thread Extraction with Decoupled Software Pipelining.* s.l. : Departments of Computer Science and Electrical Engineering, Princeton University .
10. *A Multi-Threaded Programming Strategy for Parallel Weather Forecast Model Using C#.* **Barbhuiya, Sakil and Liang, Ying.** s.l. : IEEE, 2012.
11. *A Decade of Progress in Parallel Programming Productivity .* **Richards, John, et al.** 11, s.l. : ACM, 2014, Vol. 57.
12. **Deniz, Etem and Sen, Alper.** *Using Machine Learning Techniques to Detect Parallel Patterns of Multi-threaded Applications.* s.l. : Springer Science, 2015.
13. *Drainage network extraction from a high-resolution DEM using parallel programming in the .NET framework.* **Du, Chao, et al.** s.l. : Elsevier, 2017, Jornal Of Hydrology, Vol. 555.
14. **Silva, Fabrício Alves Barbosa da and Senger, Hermes.** *Scalability Analysis of Embarassingly Parallel Applications on Large Clusters.* s.l. : IEE, 2010.
15. **Liao, Chunhua, et al.** *Semantic-Aware Automatic Parallelization of Modern Applications Using High-Level Abstractions.* 2010.
16. **G, Mattson Timothy, Sanders, Beverly and Massingill, Berna.** *Patterns for parallel programming.* s.l. : Pearson Education, 2004.

Project Proposal

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

73/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

GRADING FORM: NEW ASSESSMENT

NICHOLAS HINDER

73

ENGAGEMENT WITH LITERATURE



A good literature review, though not explicitly titled as such. A huge number of bibliography links, though relatively few works cited. Some work to do here, but a good start.

18

KNOWLEDGE AND UNDERSTANDING

- | | | |
|---|--|-----------|
|  | A good introduction – we'll have a chat about aims, objectives, hypotheses etc. but you may want to take a look into SMART in the meantime. An explicit conclusion would be good, but the concluding statements are fair | 10 |
|---|--|-----------|

COGNITIVE, INTELLECTUAL SKILLS

- | | | |
|---|---|-----------|
|  | A good, detailed outline methodology, both for research and development. Not tied to an explicit development methodology, instead outlining process – I prefer this approach. | 16 |
|---|---|-----------|

PRACTICAL SKILLS

- | | | |
|---|---|-----------|
|  | Good, valid considerations for implementation and resourcing. | 18 |
|---|---|-----------|

TRANSFERABLE SKILLS

- | | | |
|---|--|-----------|
|  | A nicely presented and fairly well-written document. | 11 |
|---|--|-----------|