Predicting Loop Parallelisability Using Supervised Learning*

Extended Abstract†

Aleksandr Maramzin‡
The University of Edinburgh
Edinburgh, Ohio
s1736883@sms.ed.ac.uk

Bjørn Franke§
The Thørväld Group
Hekla, Iceland
larst@affiliation.org

Valerie Béranger
Inria Paris-Rocquencourt
Rocquencourt, France

Aparna Patel
Rajiv Gandhi University
Doimukh, Arunachal Pradesh
India

Huifen Chan
Tsinghua University
Haidian Qu, Beijing Shi
China

Charles Palmer
Palmer Research Laboratories
San Antonio, Texas
cpalmer@prl.com

John Smith
The Thørväld Group
jsmith@affiliation.org

Julius P. Kumquat
The Kumquat Consortium
jpkumquat@consortium.net

ABSTRACT

All modern hardware is highly parallel, but in order to fully utilise availability of all these resources software must be parallel as well. There are numerous approaches to the task of software parallelisation ranging from manual approaches to fully automatic ones. In this work we investigate a relatively new semi-automated approach to the task of software parallelisation: a machine learning assisted one. We use a version of NAS Parallel Benchmarks annotated with OpenMP parallelisation pragmas to train our model to predict loop parallelisability. We achieve 93% generalised prediction accuracy with a baseline (random predictor) around 60%. We study parallelisability of NAS benchmarks and its utilisation with Intel C/C++ Compiler (ICC) and propose a scheme, which might be used to supplement ICC with a ML-based tool. Proposed scheme increases ICC parallelisation coverage in the NAS Parallel Benchmarks from 86% to 99% and comes close to its algorithmic limit. After that we show, that for majority of NAS benchmarks this parallelisation coverage increase transforms into the actual performance improvements as well. We achieve an average speedup of 2.5x relative to the state-of-the-art parallelising Intel C/C++ compiler and come closer to 3.5x average speedup of the expert hand-parallelised version.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

KEYWORDS

ACM proceedings, LaTeX, text tagging

ACM Reference Format:
Aleksandr Maramzin, Bjørn Franke, Valerie Béranger, Aparna Patel, Huifen Chan, Charles Palmer, John Smith, and Julius P. Kumquat. 1997. Predicting Loop Parallelisability Using Supervised Learning: Extended Abstract. In *Proceedings of ACM Woodstock conference (WOODSTOCK'97),* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 7 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Automatic parallelisation approaches are mainly based on a set of analyses performed statically. Static

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document
‡Dr. Trovato insisted his name to be first.
§This author is the one who did all the really hard work.

WOODSTOCK'97, July 1997, El Paso, Texas USA

ACM ISBN 123-4567-24-567/08/06. . . $15.00
https://doi.org/10.475/123_4

analyses put correctness of the resulting parallelised software at the first place. Although this conservative strategy ensures correcteness, it misses a number of opportunities, where parallelisation would also be legal and profitable. One way to deal with this problem is to supplement static analyses with dynamic information. Dynamic analyses collect an execution profile of program dependencies which materialise at the runtime and can be used to utilise a greater portion of a parallelism available in the program. Such supplement comes with the cost of actually running the program and limits parallelisation to particular program input. In our work we propose to supplement conservative static analyses with a trained parallelisability predictor instead of dynamic analysis.

Static analysis is conservative. People tackle it with dynamic profile-driven analysis and use machine learning to effectively map discovered parallelism onto different target hardware.

The *proceedings* are the records of a conference.[1] ACM seeks to give these conference by-products a uniform, high-quality appearance. To do this, ACM has some rigid requirements for the format of the proceedings documents: there is a specified format (balanced double columns), a specified set of fonts (Arial or Helvetica and Times Roman) in certain specified sizes, a specified live area, centered on the page, specified size of margins, specified column width and gutter size.

2 RELATED WORK

Machine learning techniques have already been applied in the field of compilers, but their application is mostly concentrated in the task of auto-tuning and optimal mapping of software onto a range of diverse hardware platforms. That fact has a reasonable explanaition. Compiler has to produce semantically correct translation of high-level source program into the instructions of a target machine. Statistical nature of machine learning approaches and their inherent errors limit their application in the field of compilers. Machine learning approaches has been applied to the tasks of selecting the best compiler flags for mapping software onto different kinds of hardware. This areas are safe for machine learning application.

In this work on the contrary we step into potentially dangerous machine learning application area. We investigate the task of loop parallelisability prediction. We calculate the number of unsafe error cases and propose a scheme ensuring the final correctness of produced parallel software.

This is not the only effort of extracting the best performance out of software potentially compromising its correctness. There is a vast body of research of supplementing compiler static analyses with dynamic information. These techniques have a property that they cannot generally prove correctness of profile-guided transformations for the whole range of possible program inputs.

There has already been a similar attempt to predict parallelisability of sequential programs using supervised learning methods.

3 LIMITATIONS OF THE WORK

The whole task of program parallelisation consists of the two stages: parallelism detection and its mapping onto the exact target platform. By increasing loop parallelisation coverage in the SNU NPB benchmarks we open a wider window of parallelisation opportunities on one side of the task, but it is a separate task to effectively map all these additional loops onto the whole diversity of existent parallel hardware systems. Thus in our work we do not consider the profitability of proposed additional parallelisations. Although, it is fair to say, that with our approach we fill the parallelisation gap between conservativeness of ICC compiler and OpenMP hand-annotaded benchmark versions. We use manually put parallelisation directives to train our model and compare our parallelisability increase using this manually approved parallelisation decisions against parallelising capabilities of ICC compiler. That makes it fair to assume that this increase in the number of loops to be parallelised will eventually materialise into the final performance increase, provided that the mapping is done right and effectively.

Another limitation concerns the safety of parallelisation. Like dynamic profile-guided analysis cannot completely gurantee the absence of actual dependencies for the whole variation of program inputs, our machine learning based approach introduces a fraction of false positives (non-parallelisible loops predicted to be parallelised). They constitute about 3.5% of the total number of predictions. This is the reason why we require a programmer approval for a final parallelisation decision.

4 LOST PARALLELISATION OPPORTUNITIES OF ICC COMPILER

Reduction operations on array locations .

[1] This is a footnote

5 FEATURE ENGINEERING

For any machine learning algorithm to work and make accurate predictions we need to pick the right set of quantifiable features. In our supervised learning classification task we predict parallelisability property of a loop (whether we can execute loop iterations or groups of them in parallel). We study performance of our model on the SNU NPB benchmarks. There is a range of SNU NPB loops, which escape Intel compiler parallelisation for different reasons: indirect array references, unrecognised reductions on array locations, pointers and alias analysis conservativeness, etc. But all that range of reasons is going to ultimately materialise into data and control dependencies present between loop instructions, represented as edges on the Program Dependence Graph (PDG) of a loop. Dependence relation lies at the the very essence of the loop parallelisation.

For that reason almost all features we use are based on the structural properties of the statically built Program Dependence Graph of a loop. We supplement these PDG structural features with some features capturing the nature of loop constituting instructions.

6 THE BODY OF THE PAPER

Typically, the body of a paper is organized into a hierarchical structure, with numbered or unnumbered headings for sections, subsections, sub-subsections, and even smaller sections. The command `\section` that precedes this paragraph is part of such a hierarchy.[2] LaTeX handles the numbering and placement of these headings for you, when you use the appropriate heading commands around the titles of the headings. If you want a sub-subsection or smaller part to be unnumbered in your output, simply append an asterisk to the command name. Examples of both numbered and unnumbered headings will appear throughout the balance of this sample document.

Because the entire article is contained in the **document** environment, you can indicate the start of a new paragraph with a blank line in your input file; that is why this sentence forms a separate paragraph.

6.1 Type Changes and *Special* Characters

We have already seen several typeface changes in this sample. You can indicate italicized words or phrases in your text with the command `\textit`; emboldening with the command `\textbf` and typewriter-style (for instance, for computer code) with `\texttt`. But remember, you do not have to indicate typestyle changes when such changes are part of the *structural* elements of your article; for instance, the heading of this subsection will be in a sans serif[3] typeface, but that is handled by the document class file. Take care with the use of[4] the curly braces in typeface changes; they mark the beginning and end of the text that is to be in the different typeface.

You can use whatever symbols, accented characters, or non-English characters you need anywhere in your document; you can find a complete list of what is available in the *LaTeX User's Guide* [26].

6.2 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

6.2.1 *Inline (In-text) Equations.*

A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin ... \end` construction or with the short form `$ ... $`. You can use any of the symbols and structures, from α to ω, available in LaTeX [26]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n\to\infty} x = 0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

6.2.2 *Display Equations.*

A numbered display equation—one set off by vertical space from the text and centered horizontally—is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in LaTeX; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n\to\infty} x = 0 \tag{1}$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

[2] This is a footnote.

[3] Another footnote here. Let's make this a rather long one to see how it looks.

[4] Another footnote.

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty} x_i = \int_{0}^{\pi+2} f \tag{2}$$

just to demonstrate LaTeX's able handling of numbering.

6.3 Citations

Citations to articles [6–8, 19], conference proceedings [8] or maybe books [26, 34] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation commands with a key of the item cited in the proper location in the `.tex` file [26]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the `.bib` file for your article.

The details of the construction of the `.bib` file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *LaTeX User's Guide* by Lamport [26].

This article shows only the plainest form of the citation command, using `\cite`.

Some examples. A paginated journal article [2], an enumerated journal article [11], a reference to an entire issue [10], a monograph (whole book) [25], a monograph/whole book in a series (see 2a in spec. document) [18], a divisible-book such as an anthology or compilation [13] followed by the same example, however we only output the series if the volume number is given [14] (so Editor00a's series should NOT be present since it has no vol. no.), a chapter in a divisible book [37], a chapter in a divisible book in a series [12], a multi-volume work as book [24], an article in a proceedings (of a conference, symposium, workshop for example) (paginated proceedings article) [4], a proceedings article with all possible elements [36], an example of an enumerated proceedings article [16], an informally published work [17], a doctoral dissertation [9], a master's thesis: [5], an online document / world wide web resource [1, 30, 38], a video game (Case 1) [29] and (Case 2) [28] and [27] and (Case 3) a patent [35], work accepted for publication [31], 'YYYYb'-test for prolific author [32] and [33]. Other cites might contain 'duplicate' DOI and URLs (some SIAM articles) [23]. Boris / Barbara Beeton: multi-volume works as books [21] and [20].

A couple of citations with DOIs: [22, 23].

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

Online citations: [38–40].

6.4 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material are found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed output of this document.

It is strongly recommended to use the package booktabs [15] and follow its main principles of typography with respect to tables:

(1) Never, ever use vertical rules.
(2) Never use double rules.

It is also a good idea not to overuse horizontal rules.

6.5 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

This sample document contains examples of `.eps` files to be displayable with LaTeX. If you work with

Table 2: Some Typical Commands

Command	A Number	Comments
`\author`	100	Author
`\table`	300	For tables
`\table*`	400	For wider tables

Figure 1: A sample black and white graphic.

Figure 2: A sample black and white graphic that has been resized with the `includegraphics` command.

pdfLaTeX, use files in the `.pdf` format. Note that most modern TeX systems will convert `.eps` to `.pdf` for you on the fly. More details on each of these are found in the *Author's Guide*.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper "floating" placement of tables, use the environment **figure*** to enclose the figure and its caption. And don't forget to end the environment with **figure***, not **figure**!

6.6 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. ACM uses two types of these constructs: theorem-like and definition-like.

Here is a theorem:

THEOREM 6.1. *Let f be continuous on $[a, b]$. If G is an antiderivative for f on $[a, b]$, then*

$$\int_a^b f(t)\,dt = G(b) - G(a).$$

Here is a definition:

Definition 6.2. If z is irrational, then by e^z we mean the unique number that has logarithm z:

$$\log e^z = z.$$

The pre-defined theorem-like constructs are **theorem**, **conjecture**, **proposition**, **lemma** and **corollary**. The pre-defined definition-like constructs are **example** and **definition**. You can add your own constructs using the *amsthm* interface [3]. The styles used in the `\theoremstyle` command are **acmplain** and **acmdefinition**.

Another construct is **proof**, for example,

PROOF. Suppose on the contrary there exists a real number L such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c}\left[gx \cdot \frac{f(x)}{g(x)}\right] = \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. □

7 CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LaTeX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

A HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e., the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters.

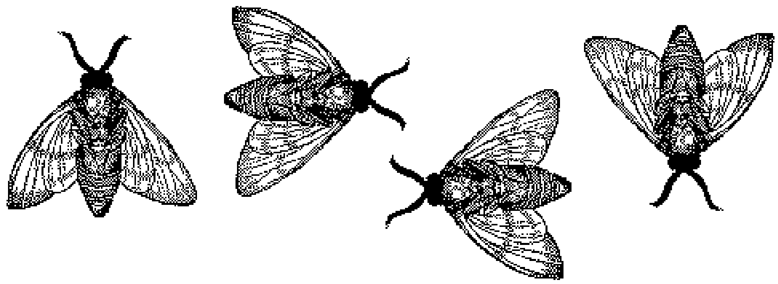

Figure 3: A sample black and white graphic that needs to span two columns of text.

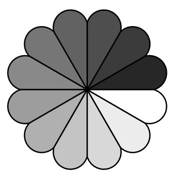

Figure 4: A sample black and white graphic that has been resized with the `includegraphics` command.

A.2.2 Math Equations.

Inline (In-text) Equations.

Display Equations.

A.2.3 Citations.

A.2.4 Tables.

A.2.5 Figures.

A.2.6 Theorem-like Constructs.

A Caveat for the TEX Expert.

A.3 Conclusions

A.4 References

Generated by bibtex from your `.bib` file. Run latex, then bibtex, then latex twice (to resolve references) to create the `.bbl` file. Insert that `.bbl` file into the `.tex` source file and comment out the command `\thebibliography`.

B MORE HELP FOR THE HARDY

Of course, reading the source code is always useful. The file `acmart.pdf` contains both the user guide and the commented code.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Yuhua Li for providing the MATLAB code of the *BEPS* method.

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is supported by the National Natural Science Foundation of China under Grant No.: 61273304 and Young Scientists' Support Program (http://www.nnsf.cn/youngscientists).

REFERENCES

[1] Rafal Ablamowicz and Bertfried Fauser. 2007. CLIFFORD: a Maple 11 Package for Clifford Algebra Computations, version 11. (2007). Retrieved February 28, 2008 from http://math.tntech.edu/rafal/cliff11/index.html

[2] Patricia S. Abril and Robert Plant. 2007. The patent holder's dilemma: Buy, sell, or troll? *Commun. ACM* 50, 1 (Jan. 2007), 36–44. https://doi.org/10.1145/1188913.1188915

[3] American Mathematical Society 2015. *Using the amsthm Package*. American Mathematical Society. http://www.ctan.org/pkg/amsthm.

[4] Sten Andler. 1979. Predicate Path expressions. In *Proceedings of the 6th. ACM SIGACT-SIGPLAN symposium on Principles of Programming Languages (POPL '79)*. ACM Press, New York, NY, 226–236. https://doi.org/10.1145/567752.567774

[5] David A. Anisi. 2003. *Optimal Motion Control of a Ground Vehicle*. Master's thesis. Royal Institute of Technology (KTH), Stockholm, Sweden.

[6] Mic Bowman, Saumya K. Debray, and Larry L. Peterson. 1993. Reasoning About Naming Systems. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 795–825. https://doi.org/10.1145/161468.161471

[7] Johannes Braams. 1991. Babel, a Multilingual Style-Option System for Use with LaTeX's Standard Document Styles. *TUGboat* 12, 2 (June 1991), 291–301.

[8] Malcolm Clark. 1991. Post Congress Tristesse. In *TeX90 Conference Proceedings*. TeX Users Group, 84–89.

[9] Kenneth L. Clarkson. 1985. *Algorithms for Closest-Point Problems (Computational Geometry)*. Ph.D. Dissertation. Stanford University, Palo Alto, CA. UMI Order Number: AAT 8506171.

[10] Jacques Cohen (Ed.). 1996. Special issue: Digital Libraries. *Commun. ACM* 39, 11 (Nov. 1996).

[11] Sarah Cohen, Werner Nutt, and Yehoshua Sagic. 2007. Deciding equivalances among conjunctive aggregate queries. *J. ACM* 54, 2, Article 5 (April 2007), 50 pages. https://doi.org/10.1145/1219092.1219093

[12] Bruce P. Douglass, David Harel, and Mark B. Trakhtenbrot. 1998. Statecarts in use: structured analysis and object-orientation. In *Lectures on Embedded Systems*, Grzegorz Rozenberg and Frits W. Vaandrager (Eds.). Lecture Notes in Computer Science, Vol. 1494. Springer-Verlag, London, 368–394. https://doi.org/10.1007/3-540-65193-4_29

[13] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago. https://doi.org/10.1007/3-540-09237-4

[14] Ian Editor (Ed.). 2008. *The title of book two* (2nd. ed.). University of Chicago Press, Chicago, Chapter 100. https://doi.org/10.1007/3-540-09237-4

[15] Simon Fear. 2005. *Publication quality tables in LaTeX*. http://www.ctan.org/pkg/booktabs.

[16] Matthew Van Gundy, Davide Balzarotti, and Giovanni Vigna. 2007. Catch me, if you can: Evading network signatures with web-based polymorphic worms. In *Proceedings of the first USENIX workshop on Offensive Technologies (WOOT '07)*. USENIX Association, Berkley, CA, Article 7, 9 pages.

[17] David Harel. 1978. *LOGICS of Programs: AXIOMATICS and DESCRIPTIVE POWER*. MIT Research Lab Technical Report TR-200. Massachusetts Institute of Technology, Cambridge, MA.

[18] David Harel. 1979. *First-Order Dynamic Logic*. Lecture Notes in Computer Science, Vol. 68. Springer-Verlag, New York, NY. https://doi.org/10.1007/3-540-09237-4

[19] Maurice Herlihy. 1993. A Methodology for Implementing Highly Concurrent Data Objects. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 745–770. https://doi.org/10.1145/161468.161469

[20] Lars Hörmander. 1985. *The analysis of linear partial differential operators. III*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. viii+525 pages. Pseudodifferential operators.

[21] Lars Hörmander. 1985. *The analysis of linear partial differential operators. IV*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. vii+352 pages. Fourier integral operators.

[22] IEEE 2004. IEEE TCSC Executive Committee. In *Proceedings of the IEEE International Conference on Web Services (ICWS '04)*. IEEE Computer Society, Washington, DC, USA, 21–22. https://doi.org/10.1109/ICWS.2004.64

[23] Markus Kirschmer and John Voight. 2010. Algorithmic Enumeration of Ideal Classes for Quaternion Orders. *SIAM J. Comput.* 39, 5 (Jan. 2010), 1714–1747. https://doi.org/10.1137/080734467

[24] Donald E. Knuth. 1997. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms (3rd. ed.)*. Addison Wesley Longman Publishing Co., Inc.

[25] David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY.

[26] Leslie Lamport. 1986. *LaTeX: A Document Preparation System*. Addison-Wesley, Reading, MA.

[27] Newton Lee. 2005. Interview with Bill Kinder: January 13, 2005. Video. *Comput. Entertain.* 3, 1, Article 4 (Jan.-March 2005). https://doi.org/10.1145/1057270.1057278

[28] Dave Novak. 2003. Solder man. Video. In *ACM SIGGRAPH 2003 Video Review on Animation theater Program: Part I - Vol. 145 (July 27–27, 2003)*. ACM Press, New York, NY, 4. https://doi.org/99.9999/woot07-S422

[29] Barack Obama. 2008. A more perfect union. Video. (5 March 2008). Retrieved March 21, 2008 from http://video.google.com/videoplay?docid=6528042696351994555

[30] Poker-Edge.Com. 2006. Stats and Analysis. (March 2006). Retrieved June 7, 2006 from http://www.poker-edge.com/stats.php

[31] Bernard Rous. 2008. The Enabling of Digital Libraries. *Digital Libraries* 12, 3, Article 5 (July 2008). To appear.

[32] Mehdi Saeedi, Morteza Saheb Zamani, and Mehdi Sedighi. 2010. A library-based synthesis methodology for reversible logic. *Microelectron. J.* 41, 4 (April 2010), 185–194.

[33] Mehdi Saeedi, Morteza Saheb Zamani, Mehdi Sedighi, and Zahra Sasanian. 2010. Synthesis of Reversible Circuit Using Cycle-Based Approach. *J. Emerg. Technol. Comput. Syst.* 6, 4 (Dec. 2010).

[34] S.L. Salas and Einar Hille. 1978. *Calculus: One and Several Variable*. John Wiley and Sons, New York.

[35] Joseph Scientist. 2009. The fountain of youth. (Aug. 2009). Patent No. 12345, Filed July 1st., 2008, Issued Aug. 9th., 2009.

[36] Stan W. Smith. 2010. An experiment in bibliographic mark-up: Parsing metadata for XML export. In *Proceedings of the 3rd. annual workshop on Librarians and Computers (LAC '10)*, Reginald N. Smythe and Alexander Noble (Eds.), Vol. 3. Paparazzi Press, Milan Italy, 422–431. https://doi.org/99.9999/woot07-S422

[37] Asad Z. Spector. 1990. Achieving application requirements. In *Distributed Systems* (2nd. ed.), Sape Mullender (Ed.). ACM Press, New York, NY, 19–33. https://doi.org/10.1145/90417.90738

[38] Harry Thornburg. 2001. Introduction to Bayesian Statistics. (March 2001). Retrieved March 2, 2005 from http://ccrma.stanford.edu/~jos/bayes/bayes.html

[39] TUG 2017. Institutional members of the TeX Users Group. (2017). Retrieved May 27, 2017 from http://wwtug.org/instmem.html

[40] Boris Veytsman. [n. d.]. acmart—Class for typesetting publications of ACM. ([n. d.]). Retrieved May 27, 2017 from http://www.ctan.org/pkg/acmart