Caldwell-t Algorithm Validation: Alternative proposal to the solution of Gupta for the sequencing of activities in processes of slender development of software with Critical Chain approach

Eldon Caldwell
Smart, Lean and Cognitive Systems Laboratory
Industrial Engineering Department
University of Costa Rica
San José, Costa Rica
eldon.caldwell@ucr.ac.cr

Abstract

Gupta algorithm was proposed back in 1972 to reduce cycle times, with very good performance results and it has been successfully used as a sequencing solution in critical activities in lean software develop processes. An alternative to this algorithm is presented with validation evidence on this document, using heuristic rules when equal selecting factors occurs. The algorithm, named Caldwell-T, can be used to solve, effectively, the possible clashes of activity that can occur in shared resources, as opposed to the traditional methodology, which encourages multitasking. Caldwell-t Algorithm obtained 78% performance effectiveness over Gupta proposal using a simulation model with Industry data. This improvement gives a big impact in software develop projects, when it is applied to critical chains and feeding buffers and signal calculations.

However, the new algorithm may yield less effective results compared with Gupta algorithm for certain random configurations of expected times of execution of activities, in approximately 20% of the cases, which allows to deduce that it is better to analyze the configuration of the multiproject environment with both algorithms, instead of putting them to compete. The results of this research is highly valuable because Gupta Algorithm is powerful compare with GA solutions and other IA approaches.

Keywords
Heuristic Algorithm, Lean Software Development, sequencing, six sigma, critical chain.

1. Introduction

The development of software applications has been widely studied from several perspectives, such as: project management, resource management practices, use of code building standards, good teamwork practices, etc.

Development methodologies have evolved, from those centralized in architecture to those focused on the development of functionalities rather than whole systems.

Within the study´s approaches and the search for optimization of development methodologies, in contrast to the traditional "waterfall" or cascade, emerge the agile approaches and in particular, the development of lean software, which has gained great relevance and followers.

At the beginning of the nineties, Womack, Roos and Jones (1991) in the United States published a book that changed the way of understanding the process flow. This book entitled: The machine that changed the world, sowed the base of what has become popular as lean manufacturing (Lean Manufacturing). This concept caused such a stir in academic and business circles that it forced two of its authors (James Womack and Daniel Jones) to write a more generic description, close to an optimization school of thought and which they called "lean thinking", or ñslender thinkingò (Hamilton B.,2007).

Lean thinking has been strengthened over the years, increased by application along with more specific techniques such as Six Sigma, giving rise to very diverse derivatives: Lean Six Sigma, Lean Service Management, Lean Accounting and Lean Supply Chain Management (George M., 2003). Faced with this evolution, it was foreseeable that software development sooner or later would also have to accept this current of thought as one of its alternatives.

Jatinder N.D. Gupta algorithm (1972) was proposed in order to reduce cycle times, with very good performance results and it has been successfully used as a sequencing criteria in critical activities in ñleanò software develop processes. Actually, this algorithm is the best in its kind with better results compared than Genetic Algorithms and other IA approaches until 2017 (Dou R. et al, 2016; Yassine A. et al, 2017), taking into account simplicity versus implementation cost, profitability and number of items involved (Gupta Algorithm is designed for MXN, resources and tasks). An alternative to this algorithm is presented with validation evidence on this document, using heuristic rules when equal selecting factors occurs.

This document is organized as follow. In the first section, a quick overview of literature review is presented. Then, a section with the explanation and validation evidence of the Caldwell-t heuristic algorithm including the main facts and results. Finally, data analysis and conclusions are developed.

1. Literature Review

Mary and Tom Poppendieck (2003) made the translation of lean thinking to software development processes, applying seven categories of waste, normally found in them. They took the proposal of Taichii Ohno (1976) of the seven basic categories of waste in the flow processes and defined their version for the case of software development projects: partially performed work, unnecessary work, unnecessary functionalities, frequent changes of activity (the multitasking effect), waiting, movements, defects and underutilization of human talent. In addition, to attack this waste, the researchers proposed what they considered the medicines or principles of Lean Software Development: eliminate waste, build quality, create knowledge, defer commitments, deliver quickly, respect people and optimize the whole (Poppendieck & Poppendieck, 2009).

From these principles, optimizing the whole is expressly focused on the elimination of wasteful practices in the management of software development projects. More specifically and related to this scientific article, the allocation of programming resources in multiproject environments with shared resources, can give a great opportunity to improve the execution and profitability of these processes (Suomalainen, T., Kuusela, R., and Tihinen, M., 2015).

Computer development projects have a very special dynamic: they are in themselves, a knowledge generation project and, on the other hand, team learning (Houssem Eddine N., Olfa Belkahla D. & Khaled G.,2016). The common practice is that many programmers carry out multiple activities, many of them of great complexity and that require inspiration and team talent.

The complexity of a computer development project is multiple: construction of code at the logical level, alignment with the information requirements and the needs of the users, shared resources, variable periods of concentration and inspiration of the programmers, changes in the demands of the users, etc. A traditional approach to work in an environment governed by uncertainty is, in many cases, to work in a multitasking or "multitasking" manner.
The multitasking approach is generally very ineffective, as shown in Figure 1.

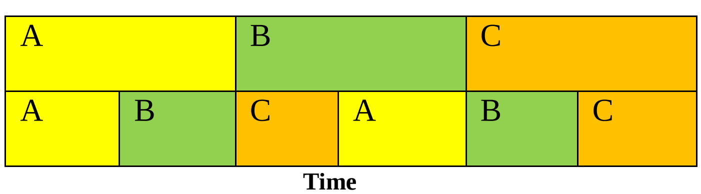

Figure 1. The multitasking effect

As can be seen in the example in Figure 1, the ABC tasks, sequenced in a dedicated manner, are completed in less time than the same tasks with a multitasking sequence. In the case of task C, there is no gain in the completion time. In specific software development projects, it is common to find that there are activities executed by shared resources. This already implies the need to determine an optimal sequence.

With much more reason, when executing multiprojects with overlapping shared resources among several projects. For example, one of the most used rules, First Incoming First Out (FIFO), is usually the immediate response to this problem (Houssem Eddine N., Olfa Belkahla D. & Khaled G.,2016).

Over the past 60 years, various heuristic algorithms have been developed to optimize the solution to the aforementioned problem:

1. Johnson Algorithm (1954-1957): 2XN, 3XN
2. Campbell Algorithm (1966-1970): MXN, N-1 sequences
3. Jatinder N.D. Gupta Algorithm (1972): MXN in two steps
4. Gupta Validation (TSung-Chyan Lai, 1996)
5. Genetic Algorithms (Chu Beasley, 1997)
6. Fuzzy algorithms and other recursive algorithms (Zimmermann, A., et al, 1996)
7. Beasley Genetic Algorithm validation (Toro E., Granada M., Romero R., 2005)
7.Caldwell-t algorithm (2010)
9. Dou R., Zong C. and Nan G. Algorithm (2016)
10.Yassine A., A, Mostafa O., Browning T.R. Algorithm (2017)

The two most robust algorithms to solve this problem are Campbell (1966) and J.N.D. Gupta (1972), and both mathematically demonstrate that the difference in the final results of service time is negligible, due to ties in the ordering criteria. Below an alternative proposal is developed to solve the problem of ordering tasks in an optimal sequence that allows the greatest use of those resources shared in critical chains in the management of software development projects and also, with the complexity of performing tasks concerned to multiple projects.

2. Caldwell-t heuristic algorithm compared with J.N.D. Gupta and Campbell algorithms

It is well known that, in an infinite number of iterations, the randomness does not generate an appreciable asymptotic limit when the criteria of the two most used algorithms are used, both Campbell and J.N.D. Gupta. But discreetly, there are differences that could be important for a particular project.
For example, let check out the series of activities related to different projects in a specialized software company located in Central America, and that they should be executed by shared resources, as expressed in Table 1.

Table 1. Sequencing problem

	Hours/Task				
Resources (for example, work stations)	A	B	C	D	E
1	66	27	80	72	64
2	28	60	14	12	92
3	38	22	26	28	74
4	2	18	100	78	6
5	76	15	100	94	81
Total	210	142	320	284	317
J.N.D. Gupta Factor	-0,025	0,0303	-0,025	-0,025	-0,0125

The Gupta algorithm is based on a discriminatory factor that is calculated as follows:

$K_i = 1 / (\text{Min} (t_{i1} + t_{i2}, \ldots\ldots. t_{in-1} + t_{in})$

Multiply k:

For 1 if t_{i1} greater than or equal to t_{in}
For -1 if t_{i1} less than t_{in}

where,

t_i is the estimated time in each activity and n is the number of activities.

After calculating K_i, the factors are ordered from lowest to highest and thus the sequence is obtained.

In this case, the sequence recommended by Gupta is: CDAEB

With this sequence, the total time of conclusion of the projects is 586 hours. As you can notice, there is a tie in the factor of activity A and C.

Considering a tiebreaker criterion, in this case, that of minimum processing time, we can propose a new algorithm, which in this case is called, Caldwell-t Algorithm (Caldwell, 2010).

This algorithm is written as follows:

1-First, an ordering factor Factor B is calculated

$B_i = 1 / (\min ((T_{i\,1} + T_{i\,2}), \ldots\ldots. (T_{i\,n-1} + T_{i\,n}))$, where $T_{i\,n}$ is the order processing time i in operation n; T_i is the processing time of the activities and n is the number of activities.

2-The B_i factor is multiplied by:

+1 if $T_{i\,1}$ greater than or equal to $T_{i\,n}$
-1 if T_{i1} less than $T_{i\,n}$

3-After assigning the sign of the B_i factor, the best sequence is ordering the obtained results from least to greatest.

4-In case of a tie between the values of the Bi factor, the work with the shortest processing time is selected. If the tie persists, the job that has the shortest cycle time between the first two operations and so on until the operation is selected is selected.

For the case presented above, we obtained that the sequence recommended by Caldwell-t algorithm is ADCEB, which results in a total project completion time (could be subgroups of complete use cases) of 554 hours against the 586 hours recommended by the Algorithm of Gupta. On the other hand, if it is compared with the Campbell Algorithm (1967), this algorithm is also robust, since Campbell throws the DCEAB sequence with a time of 564 hours.

3. Validation evidence of Caldwell-t heuristic algorithm

A fundamental need is to know if the new algorithm is statistically robust, for a very large number of iterations. The first step taken in this research was to generate 40 random iterations that would provoke ties and factor discrimination, based on data that was collected in 50 companies of the software industry in Latin America. Figure 2 shows the results.

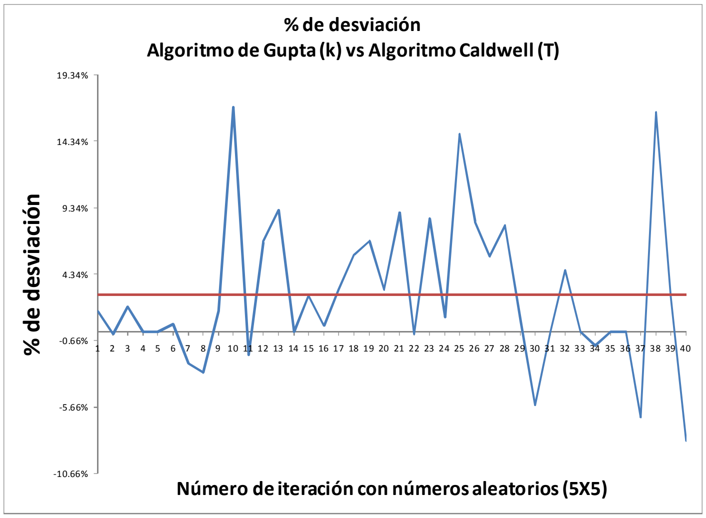

Figure 2. 40 random iterations of the Caldwell ï t heuristic algorithm.

As it is observed, there are high variations in the results, but it is notable that when the new algorithm yields shorter completion times compared to the Gupta algorithm, it does so by far. But equally, although with less probability of occurrence, when the new algorithm generates longer times against Gupta, it does so by far.

This is shown in Table 2, where it is shown that the new algorithm provides a solution equal to or better than Gupta in 78% of the cases and with a non-negligible range of variation of approximately 17%.

Table 2. Effectiveness of Caldwell-t heuristic algorithm

	Caldwell-t compared with Gupta Algorithm
Ranges	0%
	16.91%
Effectiveness	78%

If the process is simulated, but this time for a very large number of iterations, we can find a more accurate result. Using software framework tie generator, the result for 2000 random iterations generated in a standardized way is shown in table 3.

Table 3. Effectiveness of Caldwell-t heuristic algorithm for 2000 random iterations

	Caldwell-t compared with Gupta Algorithm
Ranges	0%
	18.62%
Effectiveness	78.80%

As we can see, the new algorithm provides better or equal results to the Gupta algorithm in almost 80% with a variation range of 18.62%, which shows a significant improvement. In this research, a statistical reliability test was performed, resulting in 96% confidence and an error of 3.4%.

Conclusions And Future Research Work

The Caldwell-t algorithm shows to be robust compared with Gupta's algorithm, to reduce cycle times in slender software development environments, significantly reducing waste by sequencing shared resources in multiproject and multitasking environments.

Clearly, the new algorithm could become a tool for optimizing the execution of multiple projects to execute faster the routines of small cycles of programming of use cases. It can also be used to solve, effectively, the possible clashes of activity that can occur in shared resources, as opposed to the traditional methodology, which encourages multitasking.

However, the new algorithm may yield less effective results compared with Gupta for certain random configurations of expected times of execution of activities, in approximately 20% of the cases, which allows to deduce that it is better to analyze the configuration of the multiproject environment with both algorithms, instead of putting them to compete. This is important taking into account that both algorithms have the same degree of simplicity in their calculation and application by automatic means.

With this research it is demonstrated that the development of tiebreaker rules and algorithms could generate benefits in reducing the development times of computer projects and could facilitate the adoption of lean practices.

As a future line of research, the exploration of these algorithms for different configurations of development projects with multi-routes and specific variabilities at the time of conclusion of activities is proposed.

References

Caldwell E., Lean Manufacturing: Tools and technics to reduce cycle times, Kaikaku Institute Press, USA, 2010.

Campbell, H., A heuristic technique for near optimal production schedule, Ph.D. Thesis. Texas Technological College, USA, 1966.

Campbell, H., Dudek R.A. & Smith, M.L., A Heuristic Algorithm for the n Job, m Machine Sequencing Problemò, Management Science, vol. 1, no.6, pp B630-B637, 1970.

Dou, R., Zong, C., Nan, G., Multi-stage interactive genetic algorithm for collaborative product customization. Knowledge-Based Systems, vol. 92, no.1, pp. 43-54, 2016.

Tsung-Chyan Lai, A note on heuristics of flow-shop scheduling, Operations Research, vol. 44, no. 4, pp. 648-652, 1996.

Poppendieck Mary & Poppendieck Tom, Lean Software Development: An Agile Toolkit, Addison Wesley, Indiana, USA, 2003.

Poppendieck Mary & Poppendieck Tom, Leading Lean Software Development: Results are not the point, Addison Wesley, Indiana, USA, 2009.

George M., Lean Six Sigma for Services, McGraw-Hill, Washington, USA, 2003.

Gupta, J.N.D., Heuristic Algorithms for Multistage Flow Shop Problem, *AIIE Transactions*, vol. 3, no. 4, pp 11-18,1972.

Gupta, J. N. (1971). A functional heuristic algorithm for the flowshop scheduling problem. *Journal of the Operational Research Society*, vol. *22, no.*1, pp. 39-47, 1971.

Hamilton B., Toast Kaizen. Greater Boston Manufacturing Partnership, NY, USA, 2007.

Houssem Eddine N., Olfa Belkahla D. & Khaled G., A Classification Schema for the Job Shop Scheduling Problem with Transportation Resources: State-of-the-Art Review, In book: Artificial Intelligence Perspectives in Intelligent Systems , Springer, USA, 2016.

Ohno Taichii, Toyota Production System. Productivity Press., NY,USA, 1976.

Suomalainen, T., Kuusela, R., and Tihinen, M., Continuous Planning: An Important aspect of Agile and Lean Development. International Journal of Agile Systems and Management (IJASM), vol. 8, no. 2, pp. 132ï 162, 2015.

Toro E., Granada M., Romero R. ñAlgoritmo Genético aplicado a la solución del problema de Asignación Generalizadaò. Revista Técnica UTP, vol. 8, no 16, Panama, CA, 2005.

Womack J. and Jones D., Lean Thinking, 2nd edition, Simon and Schuster, NY, USA, 2003

Womack J., Roos R., & Jones D., The Machine that changed the world, Simon and Schuster, NY,USA, 1991.

Zimmermann, A., Dalkowski K. and Hommel, G., A Case Study In Modeling And Performance Evaluation Of Manufacturing Systems Using Colored Petri Nets, 8th European Simulation Symposium (ESS '96), France, 1996.

Yassine, A. A., Mostafa, O., & Browning, T. R., Scheduling multiple, resource-constrained, iterative, product development projects with genetic algorithms. Computers & industrial engineering, vol. 107, no 1, pp. 39-56, 2017.

Biography

Eldon Caldwell, full professor (Cathedraticus) at University of Costa Rica with over 25 years of teaching and research experience, is "IEOM Outstanding Service Award" and recently selected (2018) as Fellow of the Industrial Engineering and Operations Management Society, IEOM, USA. After his Bachelor and Master degree in Industrial Engineering at University of Costa Rica, he obtained several M.Sc. degrees (MBA, Financial Analysis, Health Systems, Social Marketing, Operations Engineering, Computing Science) and finally a Ph.D. in Industrial Engineering at the University of Nevada, USA. In addition, he is Dr. Sc.(in fieri) in Automation and Robotics at the University of Alicante, Spain; and Dr.Ed.(in fieri) in Education at the University of Costa Rica, CR; and currently he is serving as Director of Industrial Engineering Department at University of Costa Rica and he is member of IEOM Society Global Council. His research interests include smart, lean and cognitive systems, robotics, cyber-physical systems and intelligent technologies for educational systems implementation in workplace for equitable employment of people with disabilities. Contact: eldon.caldwell@ucr.ac.cr