A new approach to predicting more reliable project runtimes via probabilistic model checking

Ulrich Vogl [*]   Markus Siegle [†]

1 Introduction

For more than five decades, efforts of calculating exact probabilistic quantiles for arbitrary project runtimes have not been successful due to the tremendous computation requirements, paired with hard restrictions on the available computation power. The methods established today are PERT (Programming Evaluation and Review Technique, [1, p.303-365]) and CCPM (Critical Chain Project Management, [2]). They make simplifying assumptions by focusing on the critical path (PERT) or estimating appropriate buffers (CCPM). In view of this, and since today's machines offer an increased computation power, we have developed a new approach: For the calculation of more exact quantiles or – reversely – of the resulting buffer sizes, we combine the capabilities of classical reduction techniques for series-parallel structures with the capabilities of probabilistic model checking (pMC) [4]. In order to avoid the state space explosion problem, we propose a heuristics algorithm.

2 Problem description and algorithm

The goal is to approximate the runtime distribution of a project task graph which is given as a directed acyclic graph (DAG) with one source and one sink node. Each node (task) is equipped with a stochastic execution time, given as a continuous probability distribution. Our algorithm can be outlined as follows:

- The idea is to reduce the original DAG in a stepwise fashion until the remaining graph consists of only one node whose runtime distribution approximates that of the original overall graph.
- We seek to find subgraphs which can be reduced to a single node by (exact) serial or parallel reduction, as described for example in [3]. Serial reduction means that two serially connected nodes are reduced to a single node whose distribution is the convolution of the two operand distributions. Parallel reduction means that two or more "parallel" nodes are reduced to a single node which is distributed according to the maximum of the operand runtimes.
- When no further series-parallel reduction is possible, we identify the starting and end points of a so-called complex cluster (a generally structured subgraph). We

[*] ulrich.vogl@unibw.de. Universität der Bundeswehr München, Inst. für Technische Informatik

[†] markus.siegle@unibw.de. Universität der Bundeswehr München, Inst. für Technische Informatik

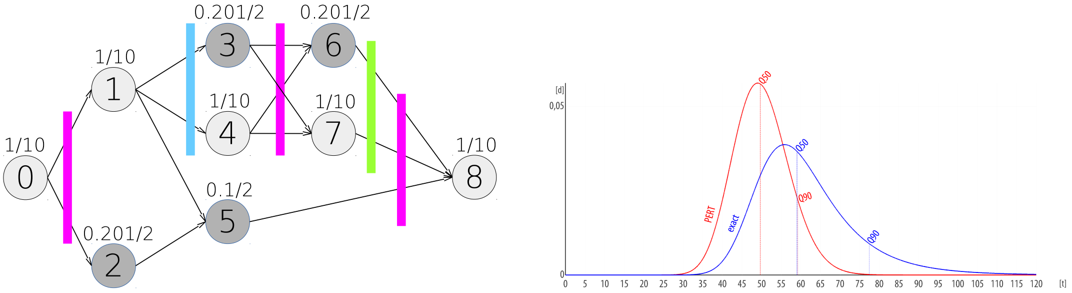
use the concept of syncpoint (see below) to define such clusters. The cluster is then reduced by a *complex reduction* to a single node. This reduction uses pMC.

- In order to avoid state space explosion, it is necessary to limit the size of the graph to be fed into pMC. Therefore the clusters analysed by pMC should be as small as possible. A related challenge consists of finding an appropriate (heuristic) fitting for each source distribution, because pMC tools (such as PRISM [5]) usually only accept exponential distributions.
- It is an interesting side effect that already one complex reduction step can often eliminate a local complexity hotspot and thereby enable further series-parallel reduction steps.

The search for appropriate starting points for cutting out clusters to be reduced can be directed by focusing on particular edge subsets: We call a set $\mathcal{E}$ of edges a *syncpoint*, if and only if for the node set P consisting of the starting points of $\mathcal{E}$ and for the node set S consisting of the end points of $\mathcal{E}$ the following holds:

1. Each edge from P to S is in $\mathcal{E}$.
2. All nodes in P have the same set of 'common' successor nodes
3. All nodes in S have the same set of 'common' predecessor nodes.

If only condition (2) with $|P| > 1$ or condition (3) with $|S| > 1$ holds, we use the concept of backward or forward half syncpoint (BHSP or FHSP).



(a) full, FH(blue), BH(green) SPs

(b) PERT calculated vs. precise quantiles

Figure 1: (a) example project graph and (b) computed runtime distributions

3 Practical implementation of the reduction steps

We use a numerical implementation of the series-parallel reduction as described in [3, p.167-226]. Furthermore, for each complex reduction we have to find an appropriate pMC model by fitting the given source distributions. For the following example we use the probabilistic Model Checker PRISM [5]. To represent each given source distribution of a complex cluster, we use the convolution of two Erlang distributions $Erl(\lambda_1, k_1) * Erl(\lambda_2, k_2)$. As long as the coefficient of variation is not greater than 1, the fitting delivers matching first two moments as well as a minimized error at the 3rd moment.

Consider the example graph in Fig. 1(a), where all nodes possess an Erlang distribution (see individual λ/k values at the nodes). Fig. 1 (b) shows the density of the PERT approach (critical path 0-1-4-7-8 only) as well as the exact target distribution of the entire graph (calculated by our method). It shows how substantial the deviation caused by the PERT-based simplification is: a confidence level of 50% is reported as 90%!

But what about CCPM? Table 1 describes all six possible paths through the graph, each equipped with an associated (feeding) buffer. The Cut & Paste Method (C&PM) [2] delivers the better quantile (84.23% at 72.53 time units), although even there remains a remarkable distance to the desired 90% quantile (appearing at 77.47 time units). Applying the Root Square Error Method (RSEM) [6, p.25ff] we get a still worse result (80.13% quantile at 70.00 time units).

Table 1: CCPM runtimes: path-wise (+buffers), maximum, corr. precise quantile

	0-2-5-8	0-1-5-8	0-1-4-7-8	0-1-4-6-8	0-1-3-7-8	0-1-3-6-8	maximum	quantile
C&PM	44.48	45.80	**48.35**	47.03	47.03	45.71	**72.53**	**84.23**%
feed.buff.	+22.24	+22.90	**+24.18**	+23.52	+23.52	+22.86		
RSEM	**44.48**	45.80	48.35	47.03	47.03	45.71	**70.00**	**80.13**%
feed.buff.	**+25.52**	+23.47	+10.15	+14.27	+14.27	+17.44		

4 Summary and future work

The presented idea offers a remarkable chance to improve the accurancy of established project planning methods. Estimates of the probable runtime – even for complex structures – can be calculated more precisely and also – compared to the customary simulation-based approaches – with manageable computation effort. One of the currently leading management methods, CCPM, can be improved by the combined use of exact calculations and heuristic approximations: For a given project schedule, one obtains a handy calculation of the time-to-finish distribution. In the opposite direction – given a desired project-finalization quantile – one gets a new, better founded calculation method for the CCPM-specific buffer dimensioning. This holds even if the scheduling complexity of the CCPM is increased by an additional calculus regarding the resources- or skill-dependencies (already envisaged in our work plan). Eventually we will use the presented method to solve resource conflicts of the kind “bad multitasking” [2] by taking or hedging a founded decision for a particular prioritization.

References

[1] Shtub, Avraham et al. (2014). *Project Management: Processes, Methodologies and Economics (2nd edition).* Pearson Education Limited, Essex.

[2] Goldratt, Eliyahu M. (1997). *Critical Chain.* The North River Press Publishing Corporation, Great Barrington.

[3] Klar, Rainer et al. (1995). *Messung und Modellierung paralleler und verteilter Rechensysteme.* B.G. Teubner, Stuttgart

[4] Kwiatkowska, Marta et al. (2007). *Stochastic Model-Checking. Vol. 4486 of Lecture Notes in Computer Science: Formal Methods for Performance Evaluation.* Springer, Heidelberg

[5] Parker, Dave et al. (2016). *The PRISM Model-Checker website.* http://www.prismmodelchecker.org

[6] Dilmaghani, Farhad (2008). *Critical chain project management (CCPM) at Bosch Security Systems (CCTV) Eindhoven: a survey to explore improvement opportunities in the scheduling and monitoring of product development projects, S. 25f..* Master Thesis, University of Twente