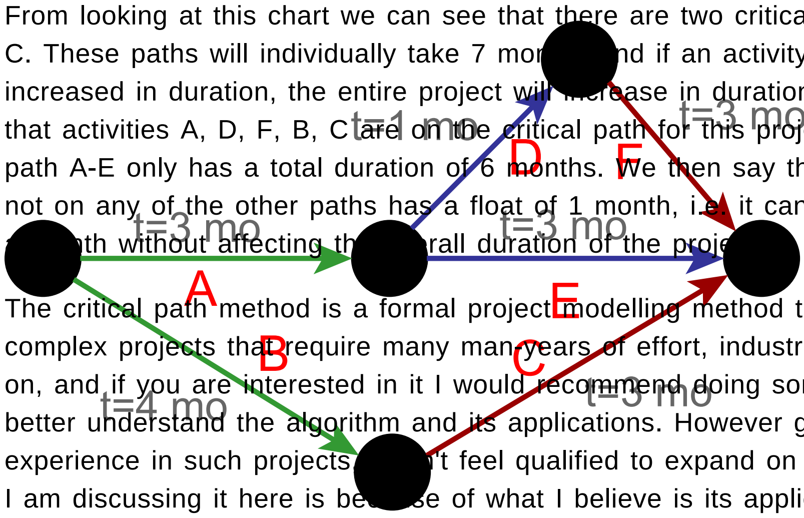
written by Thomas Maroulis on 28 March 2017 in development process, engineering

critical path method

During a previous post we started talking a bit about project management and specifically about using milestones as a way to measure progress. We also hinted at using the critical path method to detect bottlenecks in a project and this is what I want to expand on in this post.

Succinctly the critical path method is a project modelling algorithm that given an input of distinct project activities, their expected or known durations and their dependencies to each other calculates which activities form the bottleneck, i.e. which activities if sped up will improve the overall time of the entire project.

To better explain, let's look at an example. The following is a PERT chart of a project with five milestones (10-50) and six activities (A-F). Each activity has a duration of how long it is expected to take and the arrows along with the milestones indicate dependencies between activites.



From looking at this chart we can see that there are two critical paths, A-D-F and B-C. These paths will individually take 7 mon[illegible]nd if an activity on either of them is increased in duration, the entire project wil[illegible]ncrease in duration as well. That means that activities A, D, F, B, C are on the critical path for this project. Conversely the path A-E only has a total duration of 6 months. We then say that activity E which is not on any of the other paths has a float of 1 month, i.e. it can be delayed by up to [illegible]nth without affecting th[illegible]rall duration of the proje[illegible]

The critical path method is a formal project modelling method that is used to model complex projects that require many man-years of effort, industrial pipelines, and so on, and if you are interested in it I would recommend doing some further reading to better understand the algorithm and its applications. However given I do not have experience in such projects[illegible]'t feel qualified to expand on it. Instead the reason I am discussing it here is be[illegible]se of what I believe is its application in smaller projects.

The core idea behind it is that every time you have a set of independent tasks with mutual dependencies there is going to be some tasks which are on the critical path and which will directly affect the project's time estimates and some that will not. Correctly identifying what these tasks are (even if not with actual math) is a valuable first step to good prioritisation.

Doing everything at the same time has never worked for anyone and assigning the same priority to every task is axiomatically the same as saying nothing is a priority. Instead, thinking of a project in terms of dependencies and critical paths allows you to determine what can wait and for how long and what can't. This is a line of thinking that can be applied at any scale even down to the granularity of day to day planning. When you are prioritising task A over task B because task A is blocking your coworker then what you are doing is applying the critical path method. We do this because it intuitively makes sense to ensure that two people are working at the same time instead of one having to wait. What I would urge from all of this is to be explicit and methodical in applying this line of reasoning instead of only doing it intuitively. That doesn't mean doing math, but it does mean making a conscious and explicit choice to think along these lines.

That's it for this week. I hope this was useful and/or interesting.

If you enjoyed the read, drop us a comment below or share the article, follow us on Twitter or subscribe to our #MetaBeers newsletter. Before you go, grab a PDF of the article, and let us know if it's time we worked together.