Do

02.02.17

Problem: Given the project description and a number m of workers, determine which worker should perform which task in which unit of time, such that the project is completed as quickly as possible („minimize the make span")

Formally: Given directed acyclic graph $G(V,E)$, $m \in \mathbb{N}$ we want to compute a <u>schedule</u>

such that

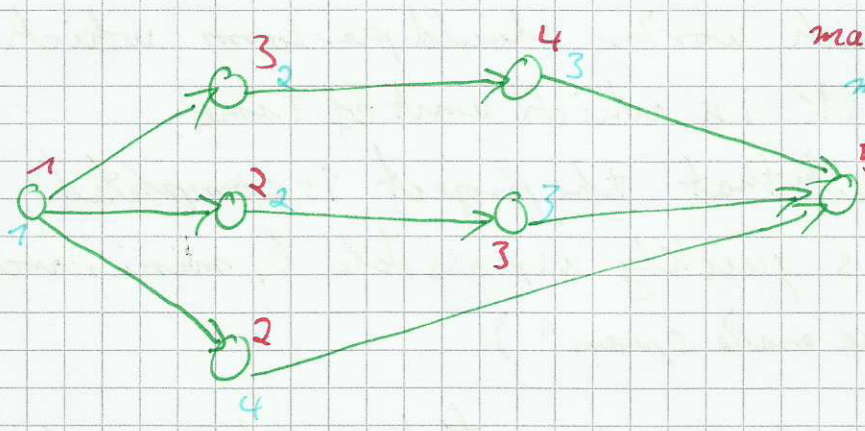
- $|\{v \in V \mid \rho(v) = i\}| \leq m$ (at no point in time more than m tasks can be worked on)
- $\forall (v,w) \in E \quad \rho(v) < \rho(w)$ (task w can only be performed after task v)

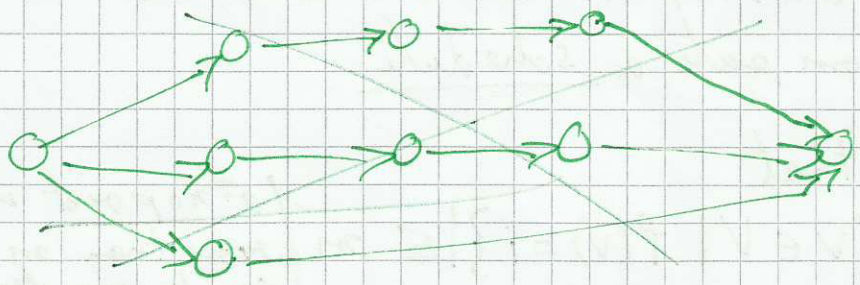
and $\max_{v \in V} \rho(v)$ is minimized

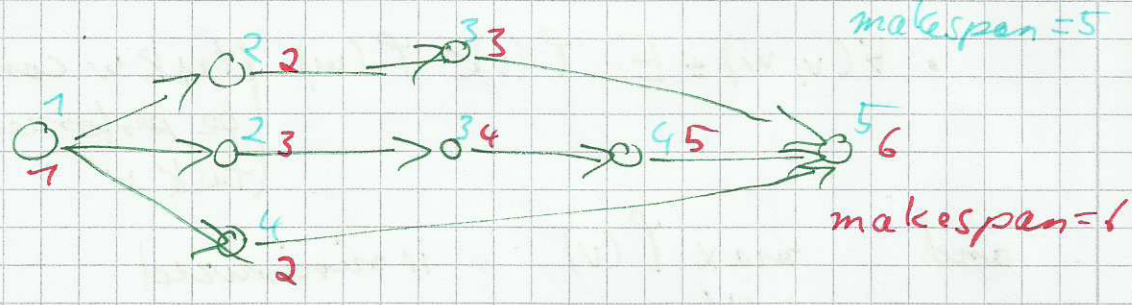
($\max_{v \in V} \rho(v)$: latest time a task is being worked on)

Example

$m = 2$







Algorithm:

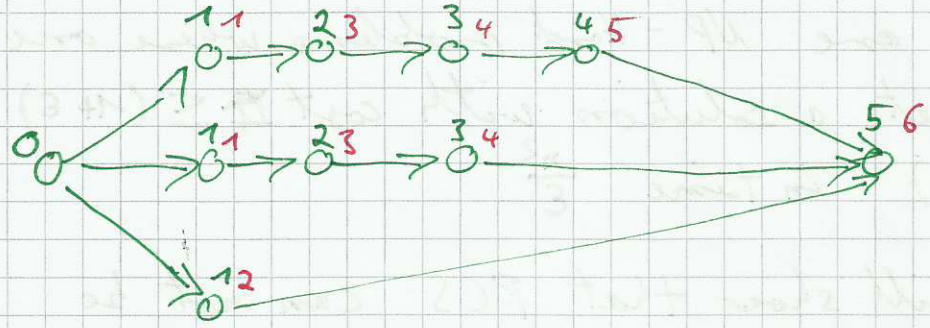
1. Add node v_0 which has an edge to all other nodes ("kickoff meeting")
2. Compute $\forall v \in V$ longest path distance from v_0 to v; call this the distance label of v

~~3. Use the first~~

$d_i := \#$ nodes with distance label i

3. use the first $\lceil \frac{d_1}{m} \rceil$ time units to process all nodes with distances label 1

- use the next $\lceil \frac{d_i}{m} \rceil$ time units to process all nodes with distance label i



Now: Prove something about the quality of the result of the algorithm

Length of produced solution (let t be the max. distance label assigned)

$$L = \sum_{i=1}^{t} \left\lceil \frac{d_i}{m} \right\rceil \leq \sum_{i=1}^{t} \left(\frac{d_i}{m} + 1 \right)$$

$$= \sum_{i=1}^{t} \frac{d_i}{m} + \sum_{i=1}^{t} 1$$

$$= \frac{1}{m} \underbrace{\sum_{i=1}^{t} d_i}_{n} + t = \underbrace{\frac{n}{m}}_{\leq L_{opt}} + \underbrace{t}_{\leq L_{opt}}$$

$$\leq 2 \cdot L_{opt}$$

Longest path distances can be computed because the graph is acyclic, i.e., there is a topological ordering of the graph.

topological: order nodes in a line such that all edges point to the right.

and we can compute longest path distances from left to right in linear time (look that up)

These are NP-Hard problems where one can get a solution with cost $\leq (1+\varepsilon) \cdot Opt$ ($\varepsilon > 0$) in time $\frac{n^2}{\varepsilon}$

We will show that PCS can not be approximated better than a factor of $\frac{4}{3}$ unless $P = NP$

We need some NP-hard problem for the reduction.

K-CLIQUE: Given a graph $G(V,E)$, $K \in \mathbb{N}$, decide ~~whether whether whether~~ whether G contains a K-Clique

K-Clique $\hat{=}$ K nodes in G which are fully connected.

Idea: Show that if PCS could be approximated better than $\frac{4}{3}$ in polynomial time, we could decide K-Clique in polynomial time

Since K-Clique is NP-hard this would show $P = NP$

DO 02.02.17/5

For a given K-Clique instance we will construct a PCS instance with following properties

- there is always a schedule of length 4
- there is a schedule of length 3 iff and only if G contains a K-Clique.

Now if we had a poly-time approximation algo approximating PCS $< \frac{4}{3}$, this algo would have to produce a schedule of length 3 if G contains a K-Clique.

$\Rightarrow$ We could decide K-Clique in poly-time.

Construction of PCS instance for $G(V,E)$ and K

Tasks $\hat{=} V \cup E \cup F_1 \cup F_2 \cup F_3$

Precedences $\hat{=}$ (v) $\rightarrow$ (e) if v is endpoint of e

and all tasks in F_i before tasks in F_{i+1}

Furthermore we choose m and $|F_i|$ as follows:

$m = K + |F_1|$

$m = |F_2| + (n-K) + \frac{K(K-1)}{2}$

$m = |E| - \frac{K(K-1)}{2} + |F_3|$

$|F_i| \geq 1$

or: choose $m = n^4$

$|F_1| = n^4 - K$

$|F_2| = n^4 - (n-K) - \frac{K(K+1)}{2}$

$|F_3| = n^4 - |E| + \frac{K(K-1)}{2}$

Observe that the instance has always has a schedule of length 4

1. step : process some K node tasks and F_1
2. Step : process remaining node tasks and F_2
3. step : process $|E| - \frac{K(K-1)}{2}$ edge task and F_3
4. Step : " remaining $\frac{K(K-1)}{2}$ edge tasks

if G has a K-Clique then there is a schedule of length 3.

1. Step: K nodes of K-Clique and T_1
2. Steps: remaining n-K nodes and all $\frac{K \cdot (K-1)}{2}$ edges of K-Clique and T_2
3. Step: remaining $|E| - \frac{K \cdot K-1}{2}$ edge tasks T_4 and T_3

If PCS instances has a schedule of length 3 there must be no idle workers at any point in time, in particular in step 2.
This is only possible if in step 1 we have processed K nodes of a Clique