Bounding series-parallel slowdown

András Z. Salamon[1,2] Vashti Galpin[3]

[1]Computing Laboratory, University of Oxford
[2]Oxford-Man Institute of Quantitative Finance
[3]LFCS, School of Informatics, University of Edinburgh

11 June 2009

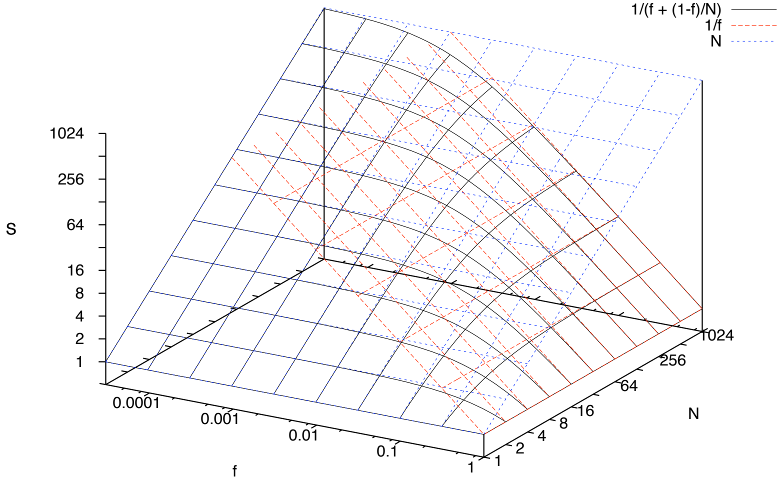
Problem

given a DAG with positive node weights, is it possible to add edges so that: the resulting DAG is series-parallel, and the slowdown ratio is at most 4/3?

Matlab Parallel Computing Toolbox
Mathematica 7
Khronos OpenCL (Apple, Nvidia, AMD, Intel)

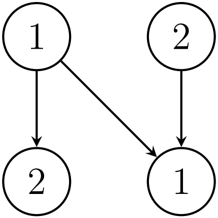
Amdahl's Law

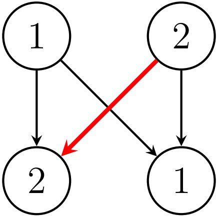
speedup S, processors N, sequential fraction f



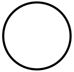
Amdahl's Law

$$1/(f + (1-f)/N) \;\to\; 1/f \quad \text{as} \quad N \to \infty$$

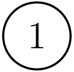

1
2
2
1


1
2
2
1

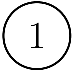
Activity network



Activity network

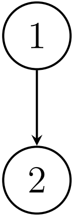


Activity network




```
a[t+1,i] = a[t,i] + sqrt(
 a[t,i-1]**2+a[t,i]+a[t,i+1]**2);
...
```

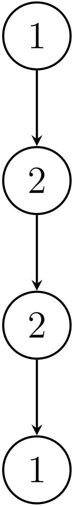
Activity network



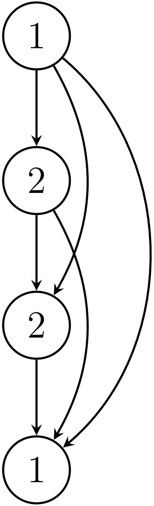

```
a[t+1,i] = a[t,i] + sqrt(
  a[t,i-1]**2+a[t,i]+a[t,i+1]**2);
...

c[t+1,i] = a[t+1,i] + ...
```

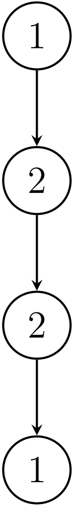
Activity network



Activity network

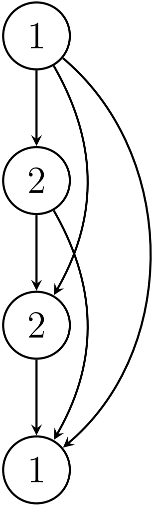


Activity network



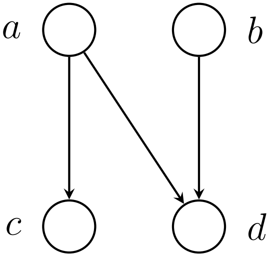
transitive reduction

Activity network



transitive closure = partial order

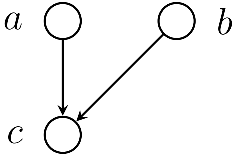
Not series-parallel



Series-parallel

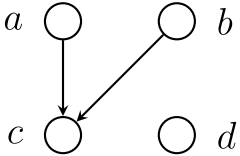
`par(a,b)`

Series-parallel



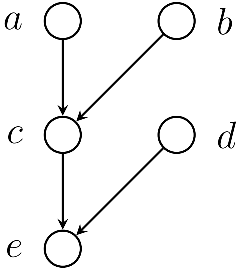
`ser(par(a,b),c)`

Series-parallel



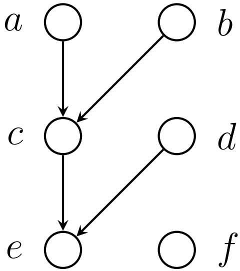
`par(ser(par(a,b),c),d)`

Series-parallel



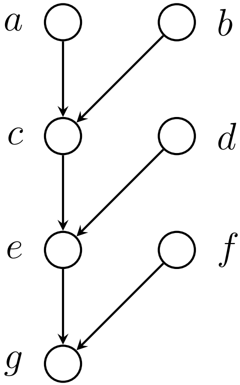
`ser(par(ser(par(a,b),c),d),e)`

Series-parallel



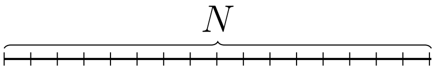

```
par(ser(par(ser(
  par(a,b),c),d),e),f)
```

Series-parallel



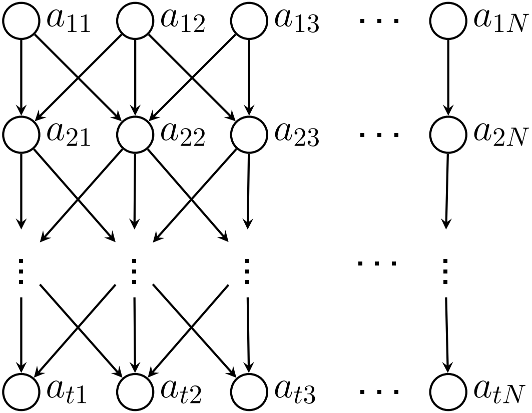

```
ser(par(ser(par(ser(
  par(a,b),c),d),e),f),g)
```

1D Flow Model

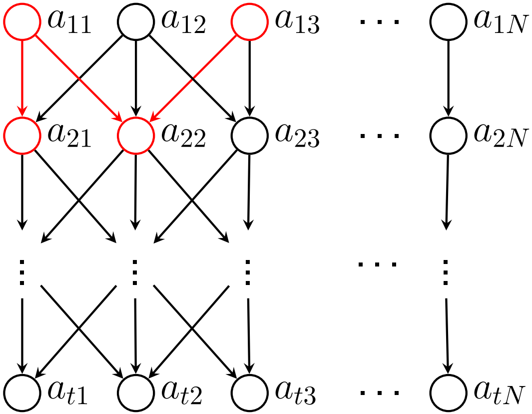


time 1: $a_{11}, a_{12}, a_{13}, \ldots, a_{1N}$
time 2: $a_{21}, a_{22}, a_{23}, \ldots, a_{2N}$
⋮
time t: $a_{t1}, a_{t2}, a_{t3}, \ldots, a_{tN}$

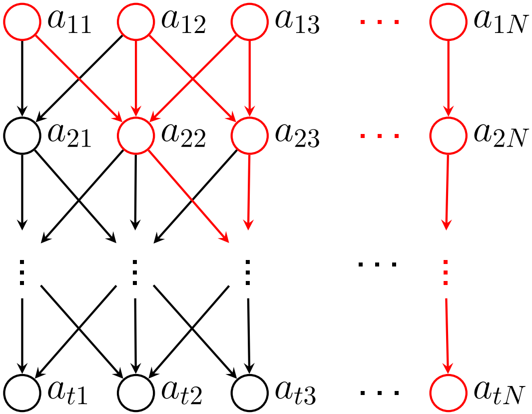
Neighbour synchronisation



Neighbour synchronisation



Pipeline



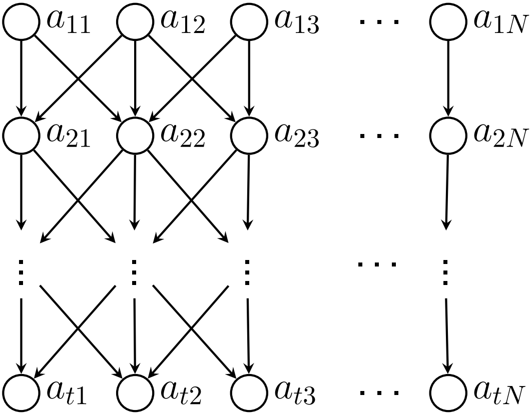
`for`

`parfor`

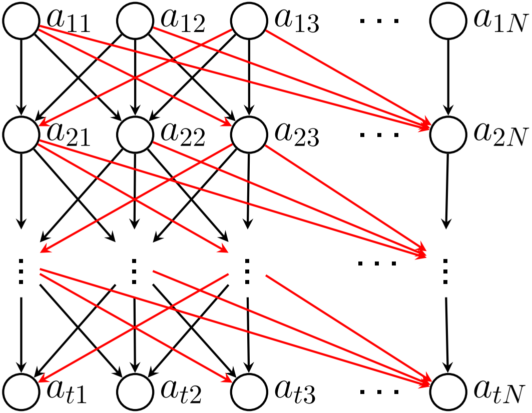
parfor

```
for i = 1 : t
    parfor j = 1 : N
        % compute a_ij
    end
end
```

parfor


a_{11} a_{12} a_{13} $\cdots$ a_{1N}
a_{21} a_{22} a_{23} $\cdots$ a_{2N}
a_{t1} a_{t2} a_{t3} $\cdots$ a_{tN}

parfor



a_{11}
a_{12}
a_{13}
$\cdots$
a_{1N}
a_{21}
a_{22}
a_{23}
$\cdots$
a_{2N}
$\vdots$
$\vdots$
$\vdots$
$\cdots$
$\vdots$
a_{t1}
a_{t2}
a_{t3}
$\cdots$
a_{tN}

$L(G)$

Let $d_G(x)$ be the depth of vertex x in poset G.

Observation $\{x \mid d_G(x) = i\}$ is independent set.

Let $L(G)$ be the ***level constrained extension*** of G, the unique poset with vertices $V(G)$ and edges E' so that for $x, y \in V(G)$

- $d_{L(G)}(x) = d_G(x)$, and
- $d_{L(G)}(x) < d_{L(G)}(y) \Leftrightarrow (x, y) \in E'$.

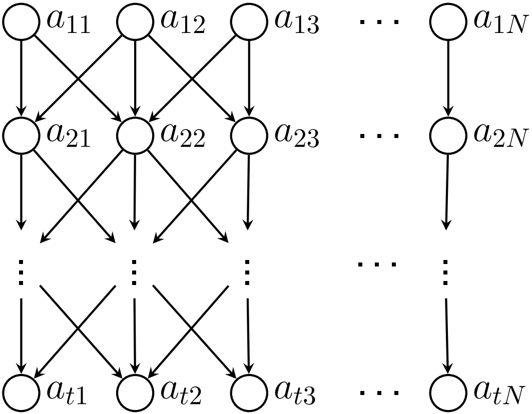
Idea due to Hu (1961).

Notation

poset	$G = (V(G), E(G))$
workload	$t\colon V(G) \to (0, \infty)$
activity network	(G, t)
duration of activity x	$t(x)$

neighbour synchronisation network
depth d, width w, degree Δ: $ns(d, w, \Delta)$

Neighbour synchronisation



$ns(t, N, 3)$

$L(G)$

$T(G,t)$ is makespan of activity network (G,t), i.e. maximum of $\sum_{x\in C} t(x)$ over all chains C in G

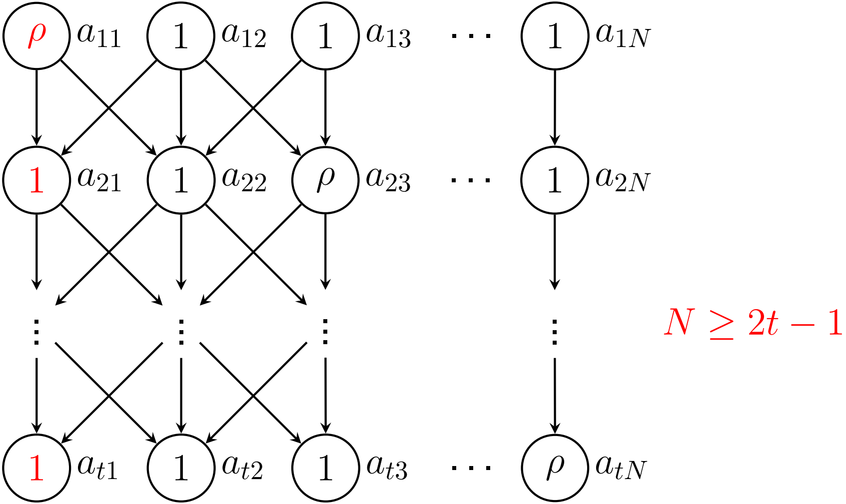
Slowdown $T(L(G),t)/T(G,t) \geq 1$

Proposition

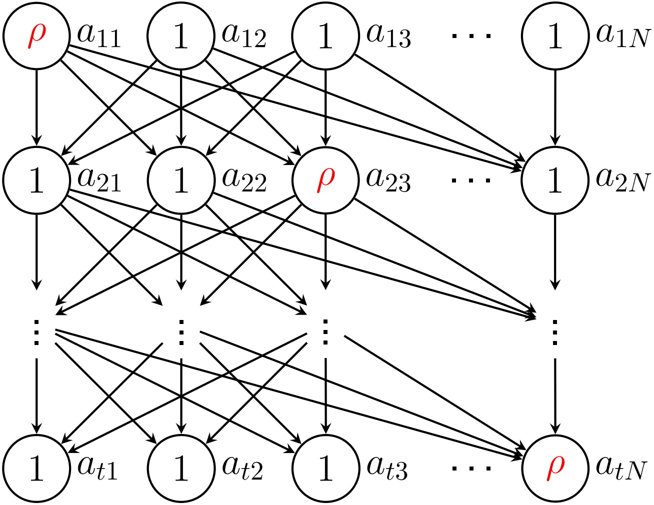
$$\frac{T(L(G),t)}{T(G,t)} \leq \frac{\max_{x\in V(G)} T(x)}{\min_{x\in V(G)} T(x)} = \rho$$

Observation This bound is (essentially) tight.

$T(ns(t, N, 3), t)$


ρ a_{11}
1 a_{12}
1 a_{13}
1 a_{1N}
1 a_{21}
1 a_{22}
ρ a_{23}
1 a_{2N}
1 a_{t1}
1 a_{t2}
1 a_{t3}
ρ a_{tN}
$N \geq 2t - 1$

$T(L(ns(t, N, 3)), t)$



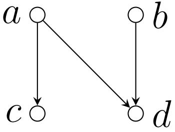
$N \geq 2t - 1$

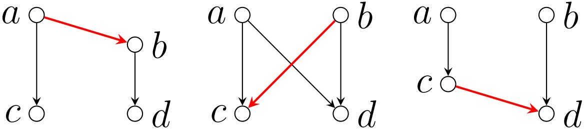
As long as $N \geq 2t - 1$,

$$\begin{aligned} \frac{T(L(G),t)}{T(G,t)} &= \frac{\rho t}{\rho + t - 1} \\ &= \rho(1 + (\rho - 1)/t)^{-1} \\ &\rightarrow \rho \quad \text{as} \quad t, N \rightarrow \infty \end{aligned}$$

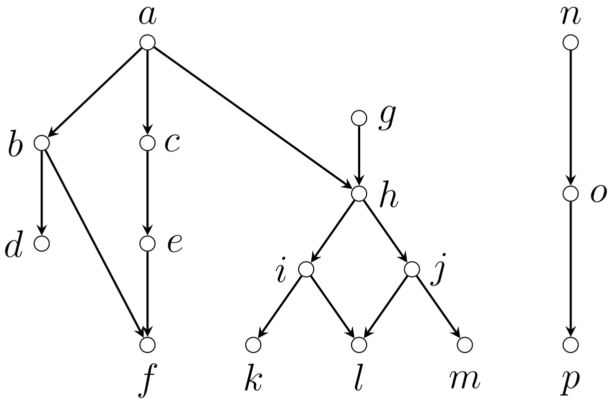
so $L(G)$ is not so good in worst case…

Three minimal choices for N

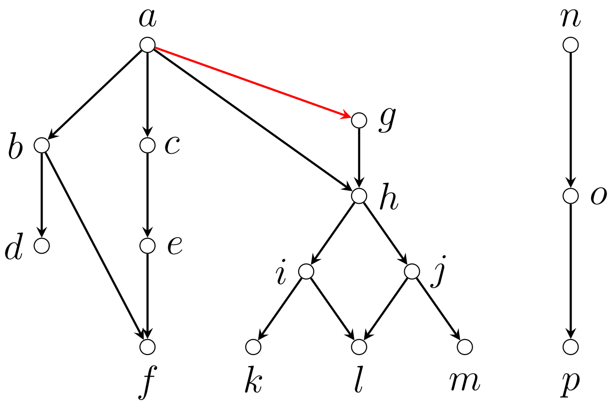




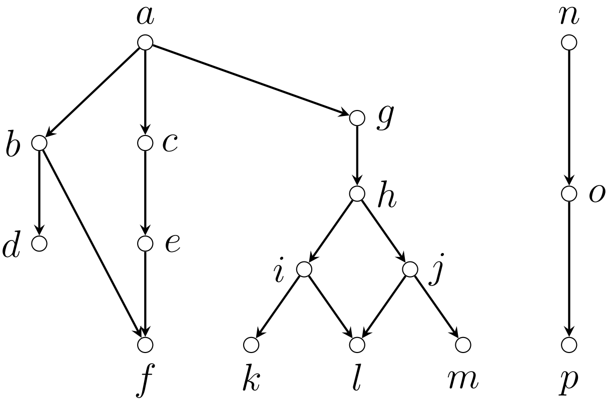
Series-parallelisation



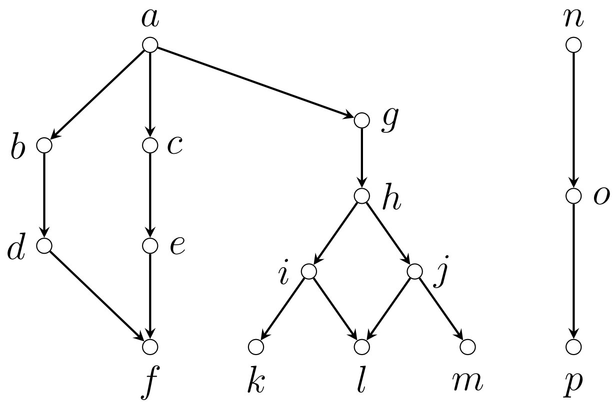
Series-parallelisation



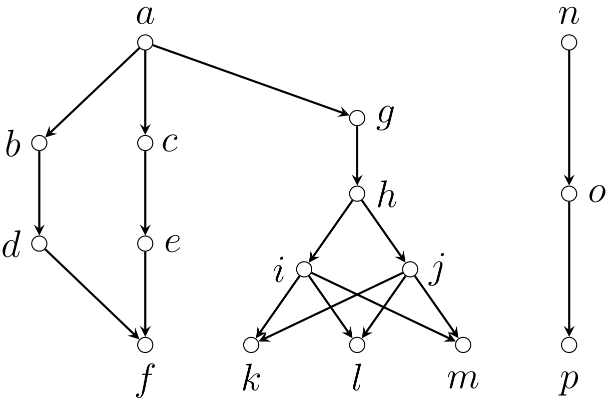
Series-parallelisation



Series-parallelisation



Series-parallelisation



Slowdown 2?

Conjecture (Van Gemund 1997) For any poset G there is G' such that for any workload t, $T(G',t)/T(G,t) \leq 2$

Algorithm, caveat: "reasonable" workloads...

G' does not use information about workload t

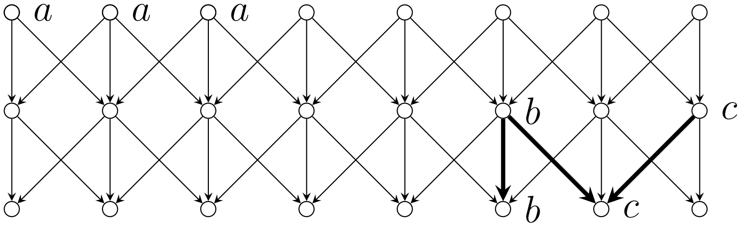
Slowdown unbounded!

outline: disprove by family of counterexamples, one for every possible series-parallelisation

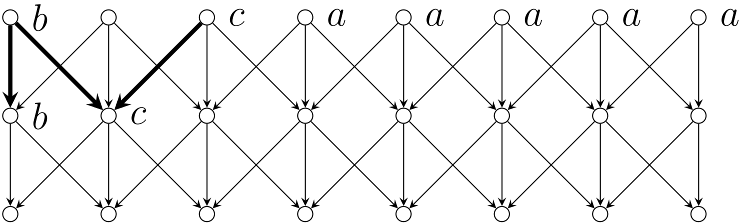
1. construct poset G and family of workloads t (each depending on SPE G') such that slowdown exceeds 2
2. for arbitrary slowdown, use copies of G as building block

"counterexample-schema"

$ns(3, 8, 3)$ case 1



$ns(3, 8, 3)$ case 2



Problem

MINIMUM SERIES-PARALLELISATION (MSP)

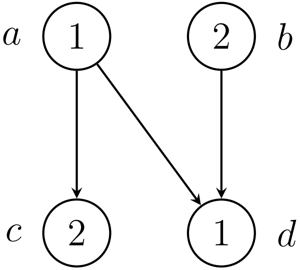
Input:	poset G, workload $t\colon V(G) \to (0, \infty)$
Output:	poset G', G' is a SPE of G
Criterion:	minimise $T(G')$

Add precedence constraints to an activity network G to obtain series-parallel G', ideally $T(G') < T(L(G))$.

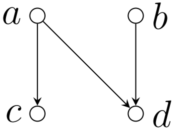
Conjecture There is constant σ such that for any activity network G there is G' with slowdown at most σ.

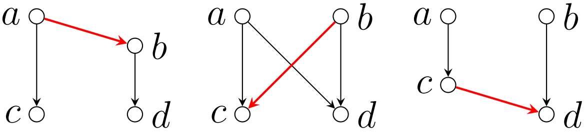
Proved for $|V(G)| \leq 6$ and $\sigma = 4/3$.

$\sigma \geq 4/3$

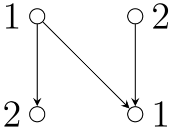


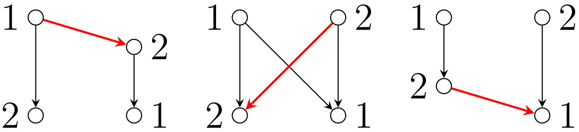
$\sigma \geq 4/3$

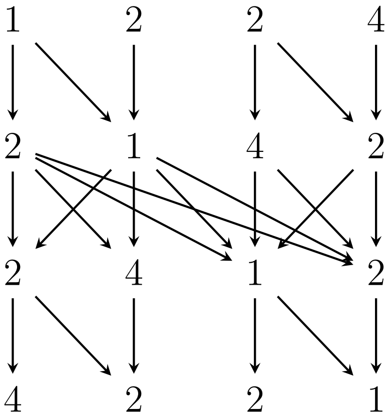




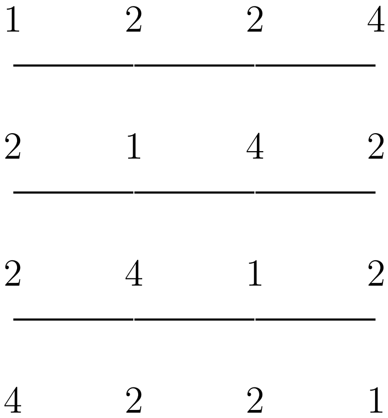
$\sigma \geq 4/3$



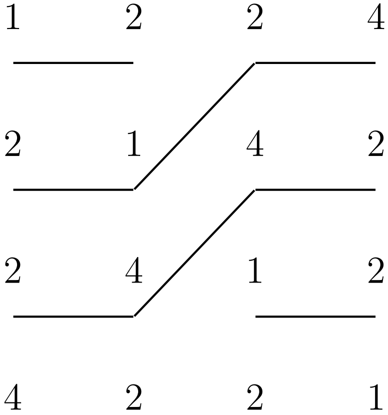




$$T(G) = 9$$



$$\frac{T(L(G))}{T(G)} = 16/9 > 4/3$$



$$\frac{T(G')}{T(G)} = 12/9 = 4/3$$

σ-MSP

Input:	activity network G
Question:	find G' with slowdown σ.

Conjecture $4/3$-MSP is NP-hard.

Different to most NP-hard problems: edge modification problems typically count number of edges added, optimization does not usually modify structure, how to capture SAT?

Related question

PRECEDENCE CONSTRAINED SCHEDULING

Input: integer N, poset G, unit durations, deadline D

Question: is there G' extension of G of width N, $T(G') < D$.

Theorem (Ullman 1975) PCS is NP-hard.

Related question

PCS2

Input: integer N, poset G, workload t, deadline D

Question: is there G' extension of G of width N, $T(G') < D$.

Conjecture PCS2 is NP-hard.

Take-home message

- series-parallel constructs force implicit precedence constraints
- G to G' results in slowdown
- simple algorithm $G' = L(G)$: unbounded slowdown
- bounded slowdown: NP-hard?
- **slowdown $4/3$ is inherent**

What to do?

- disprove conjecture, accept $4/3$ slowdown (done)
- accept yet another NP-hard problem, **on top of** scheduling, register allocation, optimal code partitioning, ...
- activities similar size: network, cache miss
- automated analysis: sometimes works well, hard!
- better languages: CCS/CSP difficult

Further work

- more precise complexity analysis of MSP
- examining scheduling to avoid this issue:
  - static: system design for similar durations
  - semi-static: build NS into language
  - dynamic: NS and other patterns auto-detected

arXiv:0904.4512