Scheduling: das Problem, mehrere (viele?) parallele Akteure optimal zu steuern

Ernst W. Mayr
Lehrstuhl für Effiziente Algorithmen
Fakultät für Informatik
Technische Universität München

Vortragsübersicht:

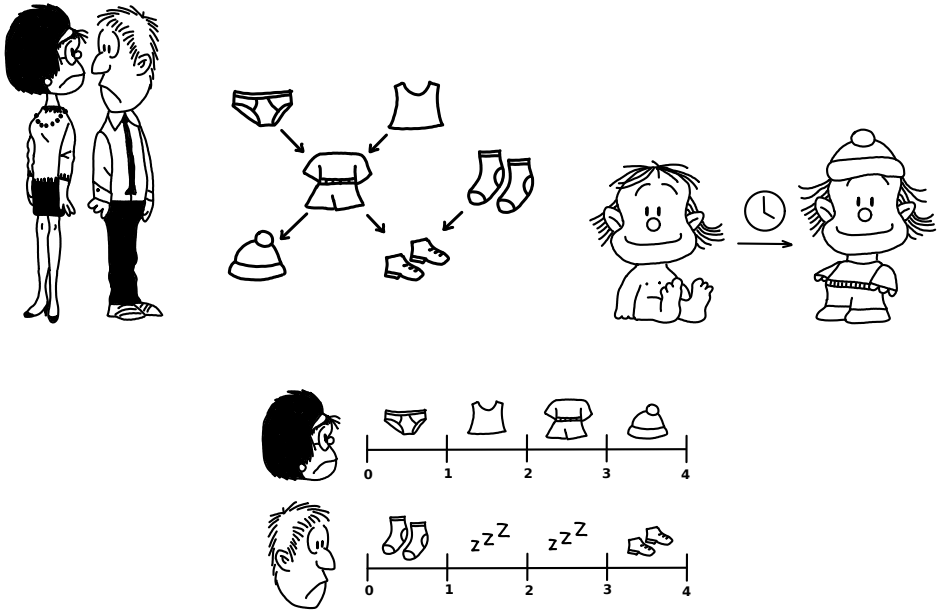
- Begriffsübersicht
- Einige einfache Beispiele
- UET und 2 Prozessoren
- Vielfalt und Komplexität
- Stochastisches Scheduling
- Einige Ergebnisse
- Zusammenfassung, offene Probleme

Was ist *Scheduling*?

- (engl. für „Zeitplanerstellung“), auch Zeitablaufsteuerung, in der Betriebswirtschaft Ablaufplanung, Maschinenbelegungsplanung oder Reihenfolgeplanung, versteht man das Erstellen eines Ablaufplanes (schedule), der Prozessen zeitlich begrenzt Ressourcen zuteilt (Allokation);
- in der Informatik im Bereich der Betriebssysteme legt Scheduling u.a. fest, welche Prozesse wann und wie viel Prozessorzeit erhalten;
- im Bereich der Datenbanktechnik wird mit dem Scheduling festgelegt, wie parallele Transaktionen ablaufen müssen, ohne die Konsistenz der Datenbank zu verletzen.

(s. Wikipedia)

Ein einfaches Beispiel



Scheduling: Was macht der PC?

- Completely Fair Scheduler
- Fair-Share-Scheduling
- Brain Fuck Scheduler (seit 2016: Multiple Queue Skiplist Scheduler)
- Highest Response Ratio Next
- Least Laxity First (Planen nach Spielraum)
- Lotterie-Scheduling
- Three-Level-Scheduling
- . . .

Beispiele

Betriebssystem	Präemption	Algorithmus
Linux kernel $<$ 2.6.0	Ja	Multilevel feedback queue
Linux kernel 2.6.0–2.6.23	Ja	O(1) scheduler
Linux kernel $>$ 2.6.23	Ja	Completely Fair Scheduler
classic Mac OS pre-9	Nein	Cooperative scheduler
macOS	Ja	Multilevel feedback queue
Windows 3.1x	Nein	Cooperative scheduler
Windows $\geq$ NT	Ja	Multilevel feedback queue

Schedulingstrategien

- Präemption
- FIFO: First in - First out, auch FCFS
- LIFO: Last in - First out
- SJN: Shortest job next, auch SJF, SPT
- EDD: Earliest due date; ähnlich EDF (earliest deadline first)
- RR: Round robin, Zeitscheibenverfahren
- HLF: Highest level first

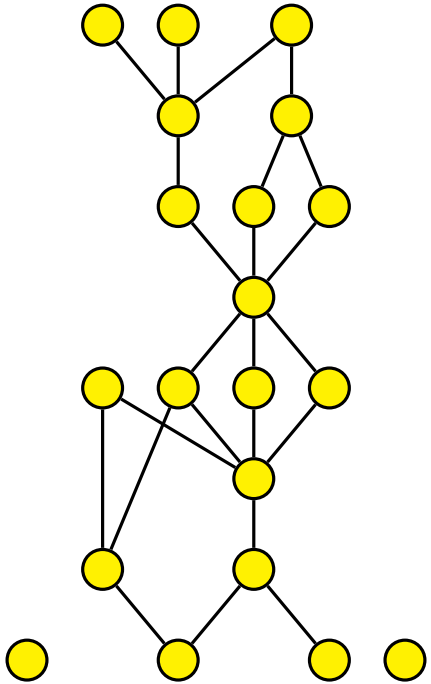
UET und 2 Prozessoren

Gegeben:

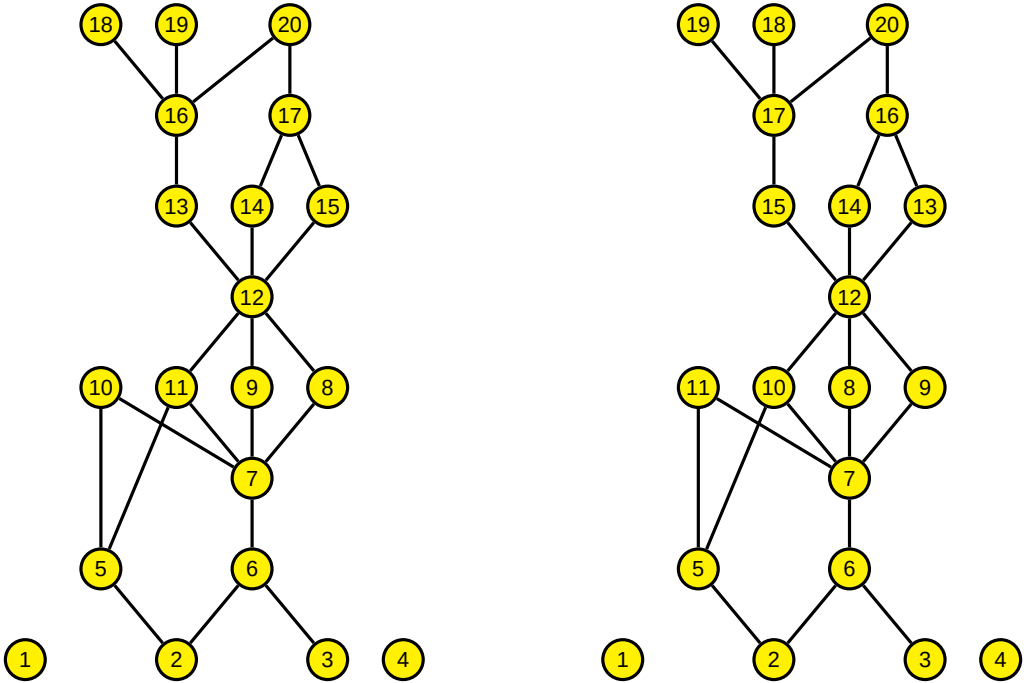
- *m* Tasks mit beliebigen Präzedenzen (Halbordnung)
- die Ausführungszeit eines jeden Tasks ist 1 Zeiteinheit (UET)
- 2 identische parallele Prozessoren
- kein (irgendwie gearteter) Overhead

Gesucht: Ein kürzester (zulässiger) Schedule!

Beispiel:

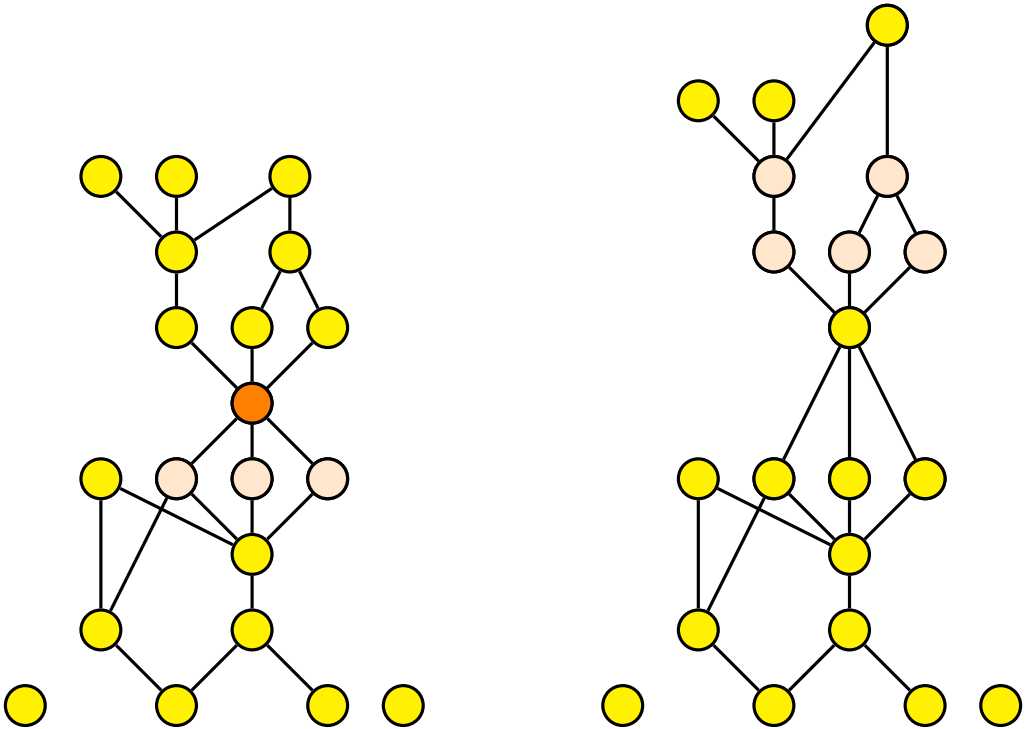


Der Coffman-Graham-Algorithmus (1972)



Zwei mögliche Ergebnisse

Zur Rettung von HLF



HLF immer optimal

Wo stehen wir?

Auf der positiven Seite:

- für UET, beliebige Präzedenzen, und #Prozessoren ≤ 2 ist HLF optimal;
- für UET, In-Tree-Präzedenzen, und #Prozessoren beliebig ist HLF optimal;
- für UET, beliebige Präzedenzen, und #Prozessoren ≤ 2 ist das Scheduling-Problem sogar in $\mathscr{NC}$ [M-Helmbold 87];
- ??

Wo stehen wir?

Auf der negativen Seite:

- für UET, beliebige Präzedenzen, und #Prozessoren beliebig vorgegeben ist das Scheduling-Problem $\mathcal{NP}$-vollständig;
- das gilt auch, wenn darüber hinaus die Problemstellung gewisse Einschränkungen enthält, wie z.B.
  - die Präzedenzordnung ist blockstrukturiert;
  - Frage ist nur: Schedule-Länge $=3$ oder mehr?;
- für beliebige (ganzzahlige) Ausführungszeiten (und die leere Präzedenz) ist das Problem $\mathcal{NP}$-vollständig;
- ??

Wo stehen wir?

Auf der “unbekannten” Seite:

- für UET, beliebige Präzedenzen, und #Prozessoren $= 3$ ist das Scheduling-Problem offen;
- und ebenso natürlich für jede #Prozessoren > 3
- . . . Und dann gibt es ja noch Unmengen anderer Varianten . . . !!

Die Vielfalt der Scheduling-Probleme

- Complex Scheduling Problems
- Scheduling Problems in Segments of a Supply Chain
- Scheduling Problems in Logistics, Transport, Timetabling, Sports, Healthcare, Engineering, Energy Management etc.
- Vehicle Routing
- Scheduling under Uncertainty
- Scheduling under Resource Constraints
- Just-in-Time Scheduling
- Assembly Scheduling
- Agent Based Scheduling
- Real-Time Scheduling
- Scheduling Heuristics and Metaheuristics
- Evolutionary Algorithms
- Approximation Algorithms
- Enumerative Algorithms
- Complexity Issues

(s. Special Issue “Algorithms for Scheduling Problems”, Algorithms (2018)

Die Vielfalt der Scheduling-Probleme II

Dazu kommen Themen, die (u.a.) an der IN.TUM (u.a.) bearbeitet werden:

- Online-Scheduling
- Randomized Scheduling
- Energy Efficient Scheduling

(s. dazu z.B. http://wwwalbers.in.tum.de/personen/albers/)

Wie sieht die Praxis aus?

Natürlich sind die meisten der Varianten des Scheduling-Problems $\mathcal{NP}$-vollständig oder $\mathcal{NP}$-hart, aber für viele lassen sich “gute” approximative Lösungen effizient berechnen:

- Berliner Busverbund
- Deutsche Bahn
- Airlines
- …

Algorithmische Ansätze und Methoden

- Relaxierung
- Integer Programming (IP), Mixed Integer Programming (MIP)
- Quadratische Programmierung, Semidefinite Programmierung
- Randomisierung, z.B. randomisiertes Runden
- . . .

Stochastisches Scheduling

Einwurf: In der Praxis sind die Ausführungszeiten der einzelnen Tasks oft nicht fest, oder sie sind unbekannt.

Man wählt daher z.B. ein Modell, in dem die Ausführungszeiten der Tasks als Zufallsvariablen dargestellt sind, mit einer (geeigneten?) Verteilung, legt eine Scheduling-Strategie fest und bestimmt die erwartete Ausführungszeit.

Dabei kann die Scheduling-Strategie deterministisch oder wiederum randomisiert sein.

Frage: Gibt es eine optimale Scheduling-Strategie?

Elementare Wahrscheinlichkeitstheorie

Wir betrachten exponentiell verteilte Zufallsvariablen $X_i \sim \mathrm{Exp}(\lambda_i)$:

- Verteilungsfunktion: $F_X(x) = 1 - e^{-\lambda x}$ für $x \geq 0, 0$ sonst
- Erwartungswert:

$$\mathbb{E}\left[X_i\right] = \frac{1}{\lambda_i}$$

- Minimum:

$$\min\{X_1, \ldots, X_n\} \sim \mathrm{Exp}(\lambda_1 + \ldots + \lambda_n)$$

- Gedächtnislosigkeit:

$$\Pr\left\{X_i > x + y | X_i > y\right\} = \Pr\left\{X_i > x\right\} \qquad (x, y > 0)$$

Snapshots

Sei m die Anzahl der verfügbaren Prozessoren. Ein Snapshot $S = (T, P, A)$ besteht aus

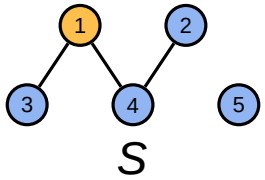
- einer Präzedenzordnung P über T,
- einer Teilmenge $A \subseteq T$ von aktiven Tasks,

wobei $|A| \leq m$ und jedes $t \in A$ eine Quelle in (T, P) ist.

Beispiel

Sei $m = 2$. Die Abbildung zeigt einen Snapshot S mit

- $T(S) = \{1, 2, 3, 4, 5\}$,
- $P(S) = \{(1, 3), (1, 4), (2, 4)\}$ und
- $A(S) = \{1\}$



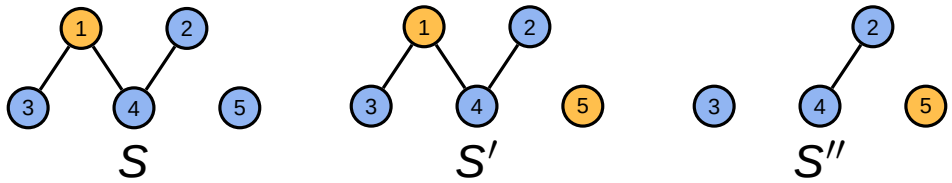
Scheduling-Strategie

- Eine Strategie σ ist eine (ggf. nichtdeterministische oder randomisierte) Funktion, die Snapshots $S = (T, P, A)$ auf Snapshots $\sigma(S) = (T, P, A')$ mit $A \subseteq A'$ abbildet.
- Ein Snapshot S', den man aus $\sigma(S)$ durch Enfernen eines Tasks $t \in A$ erhält, heißt direkter Nachfolger von S.

Beispiel

Snapshots S, S', S'' mit

- $\sigma(S) = S'$ für eine Strategie σ, und
- S'' direkter Nachfolger von S'



Die erwartete Ausführungsdauer

Unter der Annahme, dass die ZVs $X_1, \ldots, X_n \sim \mathrm{Exp}(1)$ unabhängig sind, erhalten wir für eine gegebene (deterministische) Strategie σ:

$$\begin{aligned}
\mathbb{E}[C_S^\sigma] &= \mathbb{E}\left[C_{\sigma(S)}^\sigma\right] \\
&= \mathbb{E}[\min\{X_t \mid t \in A(\sigma(S))\}] + \sum_{S' \in D(\sigma(S))} \mathbb{E}\left[C_{S'}^\sigma\right] \cdot \frac{1}{|D(\sigma(S))|} \\
&= \frac{1}{|D(\sigma(S))|}\left(1 + \sum_{S' \in D(\sigma(S))} \mathbb{E}\left[C_{S'}^\sigma\right]\right),
\end{aligned}$$

wobei $D(S)$ die Menge aller direkten Nachfolger des Snapshots S ist.

Was wissen wir?

- Für $X_i \sim \mathrm{Exp}(\lambda)$, Präzedenz In-Tree und $m = 2$ ist HLF optimal (Chandy-Reynolds'75)
- Für viele Verteilungen, Präzedenz In-Tree und *m* beliebig ist HLF "fast" optimal (Papadimitriou-Tsitsiklis'87)
- Obere Schranken sind PSPACE

Neue Ergebnisse

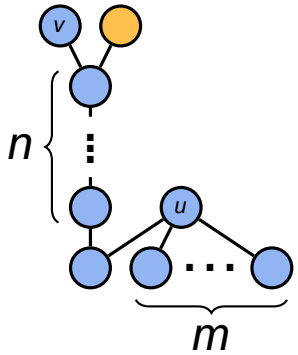
- HLF ist für In-Trees, identisch exp. verteilte Ausführungszeiten und $m = 3$ (oder mehr) Prozessoren strikt suboptimal
- HLF ist für allgemeine Präzedenzen, identisch exp. verteilte Ausführungszeiten und $m = 2$ Prozessoren strikt suboptimal
- HLF ist sogar für Out-Trees, identisch exp. verteilte Ausführungszeiten und $m = 2$ Prozessoren strikt suboptimal
- “alle anderen offensichtlichen” Scheduling-Strategien sind selbst für Out-Trees, identisch exp. verteilte Ausführungszeiten und $m = 2$ Prozessoren strikt suboptimal

Wie geht das?

Die Idee ist, den probabilistischen Berechnungsbaum für alle (relevanten) Snapshots iterativ zu berechnen.

Die Anzahl dieser Snapshots ist i.A. exponentiell, daher ist dies nur für kleine Beispielinstanzen möglich (mit dem Fortschritt der Technologie wurden und werden diese aber immer größer).

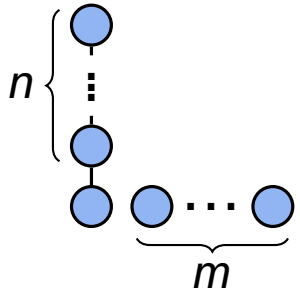
Ein für HLF problematischer Fall



Für welche Werte von m, n müsste eine optimale Strategie

- u wählen?
- v wählen?

Betrachte Snapshots $L_{m,n}$

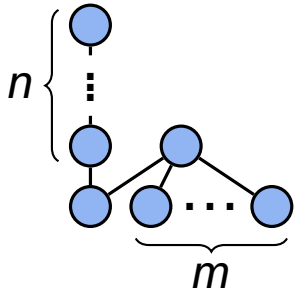


Für den optimalen erwarteten Makespan $l_{m,n}$ gilt $l_{m,0} = \frac{m}{2} + 1$, $l_{0,n} = n + 1$ und

$$l_{m,n} = \frac{1}{2} l_{m-1,n} + \frac{1}{2} l_{m,n-1} + \frac{1}{2}$$

für $m, n \geq 1$.

Für den Snapshot $H_{m,n}$

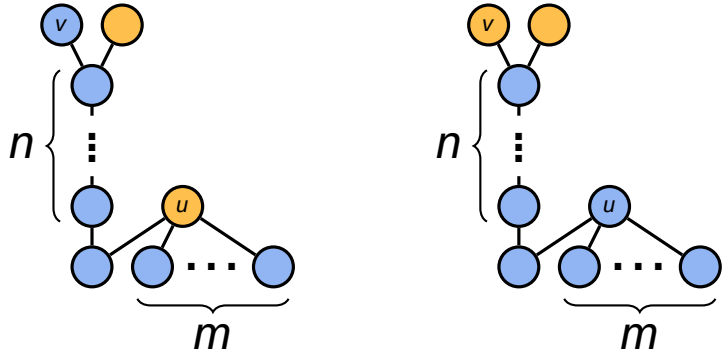


und seinen optimalen erwarteten Makespan $h_{m,n}$ gilt $h_{m,0} = \frac{m}{2} + 2$ und

$$h_{m,n} = \frac{1}{2} h_{m,n-1} + \frac{1}{2} l_{m,n} + \frac{1}{2}$$

für $m, n \geq 1$.

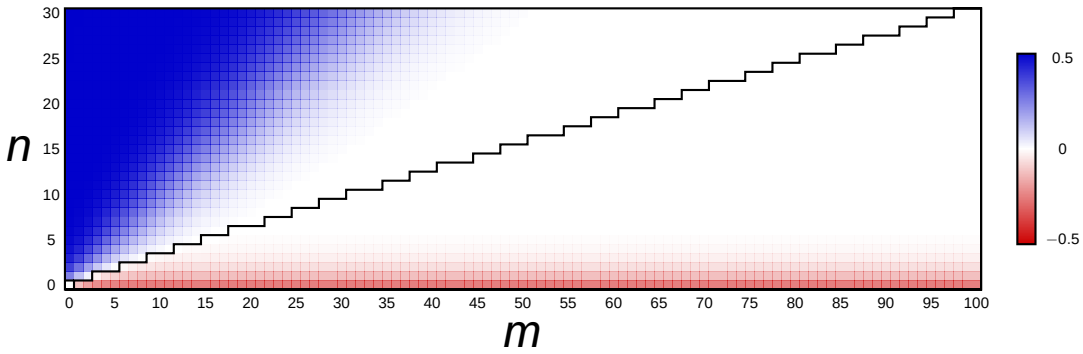
Für die Snapshots $A^u_{m,n}$ und $A^v_{m,n}$



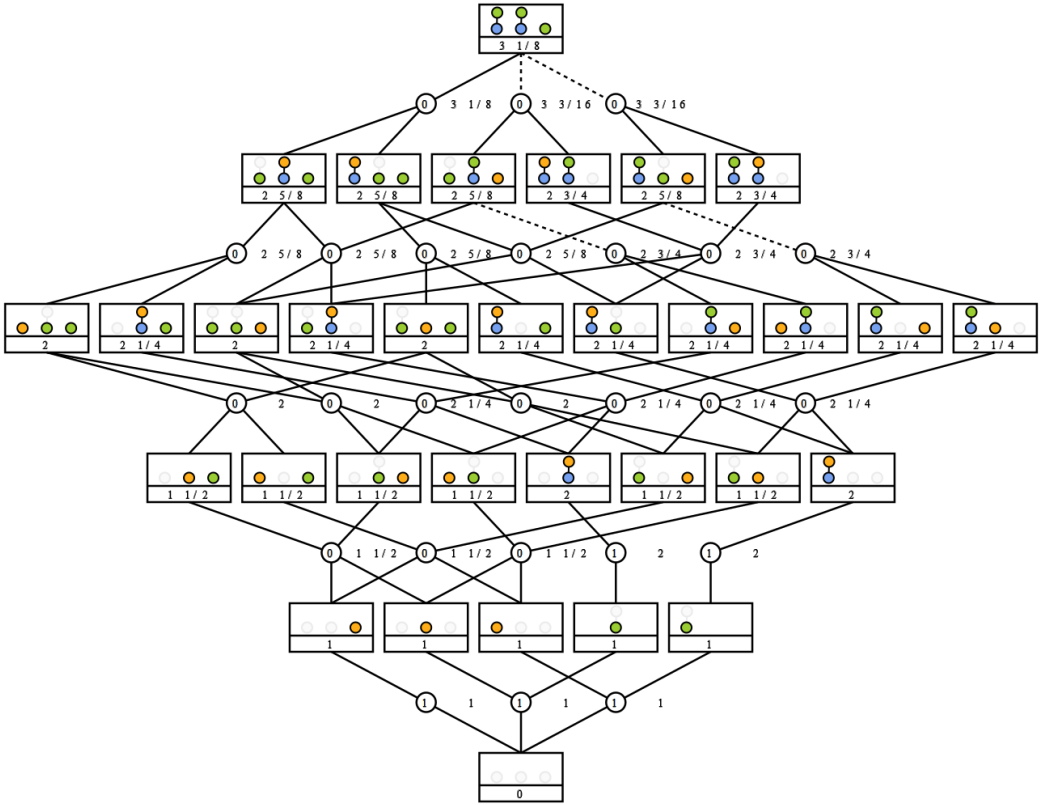
und ihre optimalen erwarteten Makespans $a^u_{m,n}$ und $a^v_{m,n}$ gilt

$$a^u_{m,n} = \frac{1}{2} l_{m,n+1} + \frac{1}{2} h_{m,n+1} + \frac{3}{4} \quad \text{und} \quad a^v_{m,n} = h_{m,n+1} + \frac{1}{2}.$$

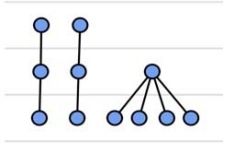
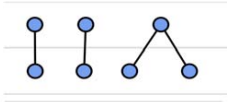
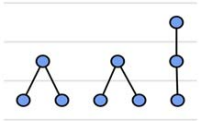
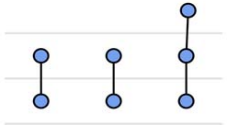
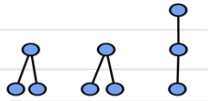
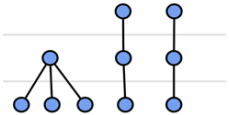
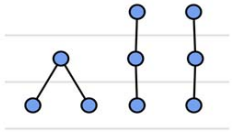
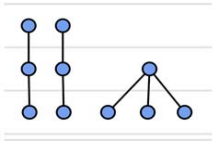
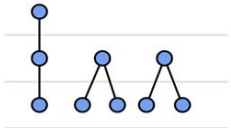
Unterschied Höhe (“n”) *vs.* Breite (“m”) $a^u_{m,n} - a^v_{m,n}$



Ein Tool:



Ergebnisse:

Strategie:	immer suboptimal	manchmal suboptimal
HLF, min pathlength, average pathlength		
immediate successors		
paths		
reachable sinks		
successors		
sum pathlengths		

Zusammenfassung:

- Scheduling, in allen seinen Varianten, ist ein wichtiges Problem
- Seine algorithmische Komplexität ist weitestgehend unverstanden
- Wir können nur (relativ) einfache Varianten effizient algorithmisch lösen
- Es gibt sehr viele algorithmische Ansätze für Plan B

Mitwirkende:

- David Helmbold (Stanford U, U Santa Cruz)
- Moritz Maaß (IN.TUM)
- Chris Pinkau (IN.TUM)
- Carlos Camino (IN.TUM, Uni Stuttgart)
- Philipp Müller (IN.TUM)
- Elashmawi Mahmoud (MA.TUM)
- Manuel Pietsch (OvT Gym Gauting)

Vielen Dank für Ihre Aufmerksamkeit!