PB165 Grafy a sítě:
Plánování pro transport, přenosy a komunikace

Obsah přednášky

1. Transport
   - Doba na nastavení
   - Doba na dopravu

2. Plánování na počítačové síti
   - Úvod k plánování úloh na počítačové síti
   - Plánování datových přenosů
   - Paralelní úlohy s komunikací

Problém obchodního cestujícího

Problém obchodního cestujícího

- obchodní cestující musí projet všechna města tak, aby
  - celková ujetá vzdálenost (resp. doba cesty) byla minimální a
  - každé město projel právě jednou

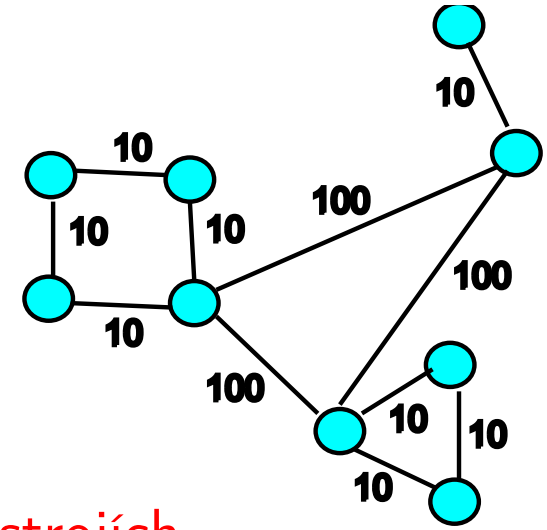
Grafová reprezentace

- (orientovaný) hranově ohodnocený graf
- vrchol = město
- (orientovaná) hrana z A do B = přímá cesta z A do B
  - hrany mohou být orientované, pokud chceme uvažovat různou náročnost v opačných směrech cesty
- ohodnocení hrany z A do B = doba nutná na cestu z A do B

Nastavovací doba a cena *(setup time and cost)*

- Nastavovací doba s_{ijk}, s_{jk}: závislá na úlohách
  - udává závislost na posloupnosti provádění úloh
  - s_{ijk} čas nutný pro provádění úlohy k po úloze j na stroji i
  - s_{jk} nastavovací doba nezávislá na stroji
- Problém obchodního cestujícího $= 1|s_{jk}|C_{\max}$
  - příklad: cesta přes města 12341 odpovídá $s_{12} + s_{23} + s_{34} + s_{41}$
- Klasický příklad: plnění limonád do lahví
  - dána cena za nastavení stroje při změně plnění typu limonády
  - $s_{cola,voda}$ $s_{voda,cola}$ $s_{cola,dzus}$
  - posloupnost plnění: 100 lahví vody, 50 lahví coly, 70 lahví džusu,
- Nastavovací cena c_{ijk}, c_{jk}
  - s přechodem lze spojit i cenu, kterou je nutne zaplatit

Doba na dopravu *(transportation time)*



Multi-operační rozvrhování

- úloha se skládá z několika operací
- může/nemusí být určeno pořadí operací
- operace má zadáno
  - dobu provádění, konkrétní stroj k provádění
- stroj: na každém stroji maximálně jedna operace
- doba na dopravu t_{hl} mezi stroji h a l: závislá na strojích
  - kapacita cest mezi stroji neomezená
  - délka cesty mezi stroji = součet odpovídajících dob na dopravu
- cíl: realizovat všechny operace všech úloh
  při minimalizaci času dokončení všech úloh

Grafová reprezentace

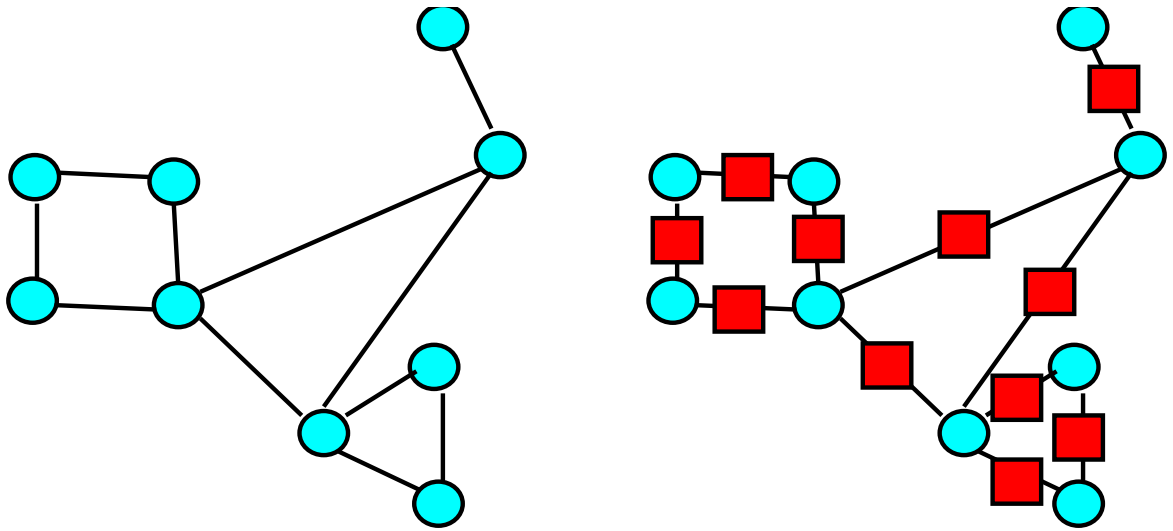
- orientovaný hranově ohodnocený graf
- vrchol: stroj
- hrana: pokud lze přejít přímo z jednoho stroje na druhý
- ohodnocení hrany: doba na dopravu z jednoho stroje na druhý

Úvod k plánování úloh na počítačové síti

- Stroj: dán počet procesorů
- Úlohy prováděny na jednom nebo více uzlech počítačové sítě
  - vyžadují několik procesorů
- Úlohy potřebují k výpočtu data
  - data dané velikosti na jednom nebo více uzlech
  - data je nutné přenést na uzel, kde se úloha bude počítat
  - realita: data jsou často zreplikována na několika uzlech
- Linka:
  - kapacita linky: velikost přenesených dat za časovou jednotku
    - např. 100Mb/s, 1Gb/s, 10Gb/s
  - latence: doba nutná na přenos dat po lince
- Cíl: realizovat všechny úlohy tak, jak dynamicky přibývají
  - úlohy musí mít dostatek procesorů
  - data musí ležet v době výpočtu na uzlu, kde se počítá úloha
  - je nutné plánovat i přenosy dat tak, aby bylo možné data přenést vzhledem k latenci i kapacitě linek na cestě

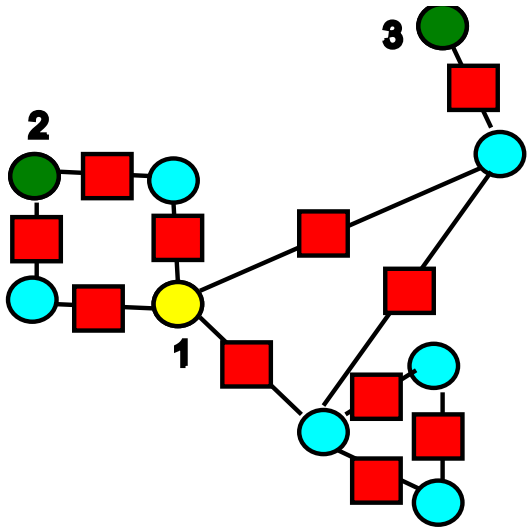
Počítačová síť: grafová reprezentace

- Vrcholově ohodnocený neorientovaný graf
- Vrchol: stroj nebo linka
- Ohodnocení vrcholu-stroje: počet procesorů
- Ohodnocení vrcholu-linky: kapacita linky
  - linka je chápána jako zdroj, který má zadánu kapacitu
  - doba zpracování úlohy na lince (tj. přenosu dat pro úlohu na lince) odpovídá latenci
- Hrany: pokud jsou stroje A a B přímo spojeny linkou C, pak existují hrany AC a BC

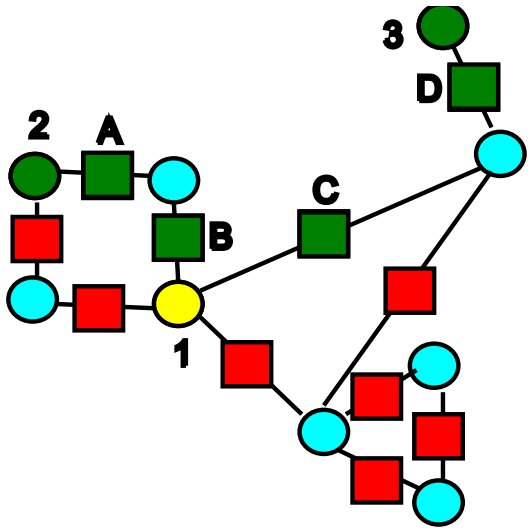


Plánování úlohy na počítačové síti: příklad

- Úloha naplánována k provádění na uzlu 1
- Data na uzlech 2 a 3

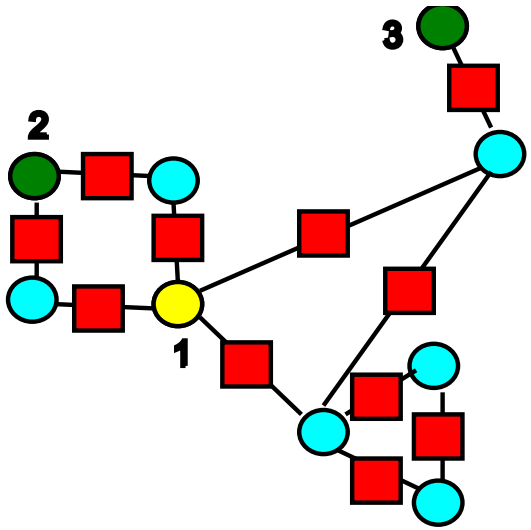


- Data jsou přenesena přes D,C a A,B
- Kapacita linek A,B,C,D musí být v daném čase postačující
- Celková doba přenostu do 1: max(latenceA+latenceB, latenceD+latenceC)

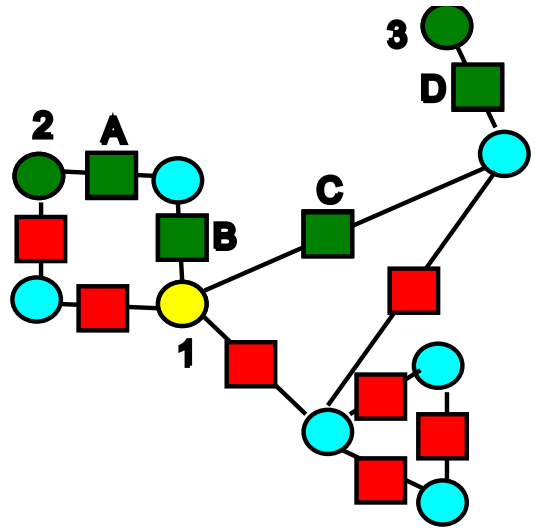


Plánování úlohy na počítačové síti: příklad

- Úloha naplánována k provádění na uzlu 1
- Data na uzlech 2 a 3



- Data jsou přenesena přes D,C a A,B
- Kapacita linek A,B,C,D musí být v daném čase postačující
- Celková doba přenostu do 1: max(latenceA+latenceB, latenceD+latenceC)



Otázky:

- Je možné takovouto úlohu naplánovat za probíhajícího provozu na síti?
- Je možné ji naplánovat při modifikaci cest pro přenosy?
- Obecně: jak naplánovat úlohu(y) za daného provozu na síť? směrování

Plánování datových přenosů

Problém:

- plánování datových přenosů, kde se
  velikost přenášených dat blíží kapacitám používaných linek

Varianty:

- plánování přenosů v čase
  - je třeba uvažovat plánování zdrojů v čase
- plánování současně probíhajících přenosů
  - není uvažován čas, vše se plánuje pro aktuální okamžik
  - není tedy ani uvažováno plánování úloh v čase na uzly
  - pro tento případ uvedeme základní model

Příklady aplikací plánování přenosů

Plánování datových přenosů pro speciální zařízení a přístroje

- umožňuje plánovat přenosy dopředu na danou dobu, kdy budou linky dostupné (bulk přenosy dat)
- např. RHIC (relativistic heavy ion collider, USA) nebo LHC (Large Hadron Collider, Evropa)

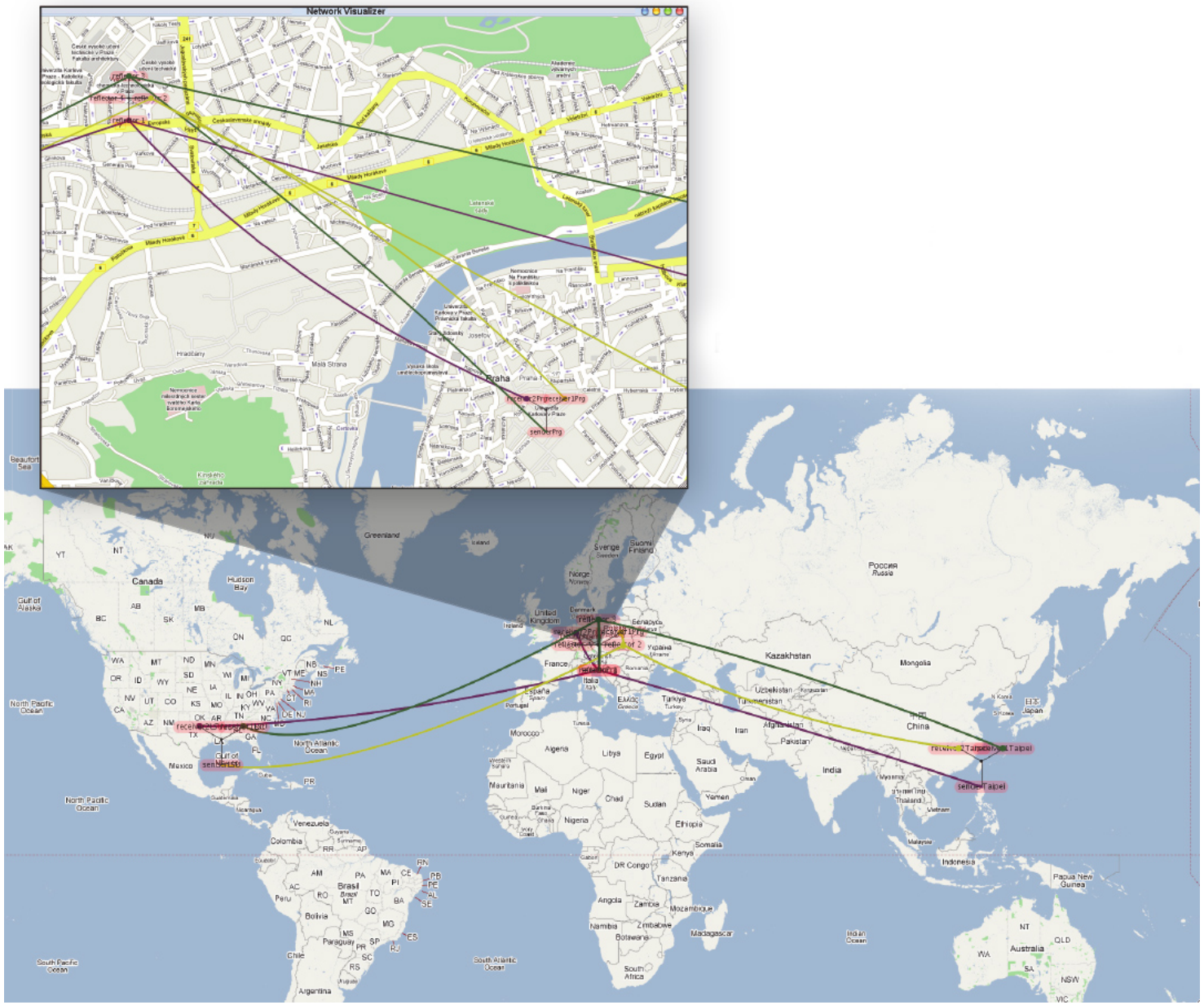
Umístění aplikací na servry

- aplikace je nutno naplánovat na servrech, plánování v daném okamžiku
- ceny: za přenos aplikací na servry, za běh aplikace na daném servru
- aplikace pro italskou mezibankovní síť

Plánování přenosů HD videa v rámci kolaborativního prostřední

- nutná reakce na okamžitý stav počítačové sítě a naplánování na síť za daných aktuálních podmínek
- např. medicínské aplikace, filmový průmysl, výuka pomocí videa (vzdálená výuka, přístup např. k operačním salům na medicíně, záznam a zpracování přednášek)
- systém CoUniverse vyvíjený na FI MU
  - `https://www.sitola.cz/CoUniverse`

Naplánované spojení pomocí CoUniverse



Grafová reprezentace

Na uzlu $v \in V$ se mohou nacházet různé aplikace

- producent $p \in P$ produkuje data
- konzument $c \in C$ přijímá data
- distributor $d \in D$ je schopen rozesílat přijímaná data do několika linek
  - pro *zjednodušení* můžeme předpokládat, že data jsou stejného typu, tj. distributor je pouze reflektorem
  - *obecně* distributor může transformovat (transkódovat) přenášená data
    - např. z nekomprimovaného HDTV videa na HDV MPEG2 video

Pro *zjednodušení* můžeme předpokládat, že na uzlu je nejvýše jedna aplikace

Uzel může mít několik rozhraní (interfaces), po kterých posílá a přijímá data

- mezi dvěma uzly může být *obecně* několik linků spojujících různá rozhraní
- pro *zjednodušení* můžeme uvažovat nejvýše jeden (orientovaný) link mezi dvěma uzly

Grafová reprezentace (dokončení)

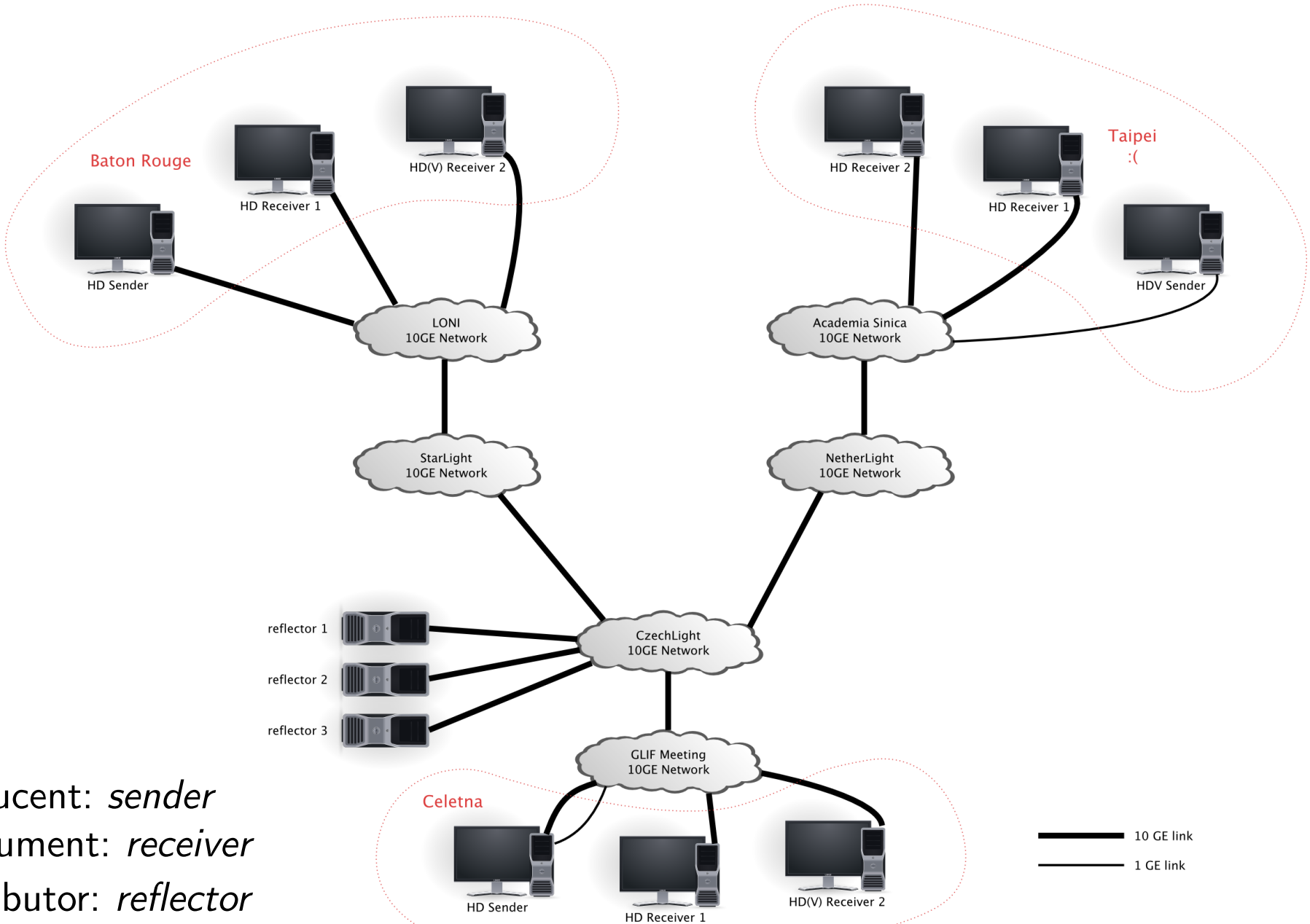
Linka $l \in E$ má určen

- počáteční uzel $begin(l)$
- koncový uzel $end(l)$
- latenci $latency(l)$
- kapacitu $capacity(l)$

Orientovaný hranově a vrcholově ohodnocený graf

- vrcholy: množina uzlů $v \in V$
- hrany: množina orientovaných linek $l \in E$
- vrcholové ohodnocení
  - producent/konzument/distributor/žádná aplikace
  - uzly bez aplikací mají smysl např. v případě změny kapacity linky
- hranové ohodnocení
  - latence + kapacita

Ukázka sítě při plánování datových přenosů



Producent: *sender*
Konzument: *receiver*
Distributor: *reflector*

Stream

Stream $s \in S$ je datový tok od producenta ke konzumentům

- $bandwidth(s)$ udává požadovanou šířku pásma datového toku streamu
- příjem dat streamu s je požadován množinou zadaných konzumentů $consumers(s) \subseteq C$
- data streamu s vysílá právě jeden producent $p = producer(s) \in P$
  - *obecně* může existovat více producentů a cílem plánování je pak vybrat producenty zajišťující maximální kvalitu danou typem přenášených dat
    - např. nekomprimované HDTV video, HDV MPEG2 video, HDV MPEG4 video
  - pro *zjednodušení* můžeme předpokládat, že každý stream má předem jednoznačně určeného producenta
    - vhodného producenta lze vypočítat předem

Problém plánování datových přenosů

Problém plánování několika streamu

- naplánovat *konzistentně* všechny streamy z S na dostupné linky tak, aby byla optimalizována zadaná kritéria, např.
  - minimalizace latence
  - při uvažování kvality spojení: maximalizace kvality přenášených dat

Model založený na lincích (*link-based model*)

- pro každý požadavek (stream) s: proměnná pro každý link l
- Proměnná $sl(s, l)$ pro každý stream s a každou linku l

$$sl(s, l) = \begin{cases} 1 & \text{jestliže } s \text{ prochází po } l \\ 0 & \text{jinak} \end{cases}$$

- Cíl: nalézt hodnoty proměnných $sl(s, l)$ optimální vzhledem k zadanému optimalizačnímu kritériu
- pro tento model si ukážeme vyjádření typických požadavků problému

Další modely pro plánování na přenosů

Model založený na cestách (*path-based model*)

- pro každý požadavek s: proměnná pro každou možnou cestu p mezi zdrojem (producent) a cílem (konzument)

$$path(s,p) = \begin{cases} 1 & \text{jestliže jde požadavek } s \text{ po cestě } p \\ 0 & \text{jinak} \end{cases}$$

- náročné pokud je hodně možných cest

Model založený na uzlech (*node-based model*)

- pro každý požadavek s: proměnná pro každý uzel v

$$var(s,v) = \begin{cases} 1 & \text{jestliže je uzel } v \text{ použit pro požadavek } s \\ 0 & \text{jinak} \end{cases}$$

Cíl: nalézt hodnoty proměnných modelu optimální vzhledem k zadanému optimalizačnímu kritériu

Vlastnosti linek

Data streamu s prochází po lince l, pokud je

- na jejím začátku producent $producer(s)$ nebo distributor $d \in D$
  tj. na začátku nesmí být ani konzument ani jiný producent než $producer(s)$

$$\forall s \in S \; \forall l \in E$$
$$(\exists c \in C : c \in begin(l)) \; \vee \; (\exists p \in P : p \neq producer(s) \wedge p \in begin(l)) :$$
$$sl(s, l) = 0$$

- na jejím konci konzument z $consumers(s)$ nebo distributor $d \in D$
  tj. na konci nesmí být ani producent ani jiný konzument než $consumers(s)$

$$\forall s \in S \; \forall l \in E$$
$$(\exists p \in P : p \in end(l)) \; \vee \; (\exists c \in C : c \notin consumers(s) \wedge c \in end(l)) :$$
$$sl(s, l) = 0$$

Požadovaná šířka pásma pro všechny streamy nesmí překročit kapacitu žádného linku

$$\forall l \in E \quad \sum_{s \in S} sl(s, l) \times bandwidth(s) \leq capacity(l)$$

Vlastnosti konzumenta a producenta

Konzument c přijímá data právě jednou linkou

$$\forall c \in C \sum_{s \in S} \sum_{l \in E : c \in end(l)} sl(s, l) = 1$$

Producent p posílá data právě jednou linkou

$$\forall p \in P \sum_{s \in S} \sum_{l \in E : p \in begin(l)} sl(s, l) = 1$$

Vlastnosti distributora

Počet linek, kterými distributor d přijímá data streamu s

$$inLinks_{sd}(s,d) = \sum_{l \in E : d \in end(l)} sl(s,l)$$

Počet linek, kterými distributor d vysílá data streamu s

$$outLinks_{sd}(s,d) = \sum_{l \in E : d \in begin(l)} sl(s,l)$$

Distributor d slouží pro distribuci dat pouze jednoho producenta,
tj. přijímá data nejvýše jedním linkem

$$\forall d \in D \quad \sum_{s \in S} inLinks_{sd}(s,d) \leq 1$$

Distributor d je schopen distribuovat data ze streamu s,
která přijímá, do několika linek

$$\forall s \in S \; \forall d \in D \quad \begin{array}{ll} \text{jestliže } inLinks_{sd}(s,d) = 0 & \text{pak } outLinks_{sd}(s,d) = 0 \\ \text{jestliže } inLinks_{sd}(s,d) = 1 & \text{pak } outLinks_{sd}(s,d) \geq 1 \end{array}$$

Optimalizační kritéria

Celková latence musí být minimalizována

$$\text{minimize} \sum_{s \in S} \sum_{l \in E} sl(s, l) \times latency(l)$$

Další kritéria:

- maximalizace kvality přenosu
  - při zajišťování linek s maximální kapacitou pro přenášená data
- balancování latencí všech linek
  - při video-konferencích k zajištění bezproblémového přerušení a vstupu do probíhající video-konference

Metody řešení

- Uvedený model popisuje základní množinu požadavků pro řešení problémů plánování datových přenosů
- Při řešení konkrétního problému nutné přidat specifická omezení

  a také redundantní omezení pro úspěšné/efektivní řešení problému
- Daný model (a jeho rozšíření na konkrétní aplikaci) lze řešit pomocí
  - grafových algoritmů (toky v síti, hledání cest v grafu)
  - programování s omezujícími podmínkami
    - např. CoUniverse využívá pro plánování
      Java řešič Choco pro omezující podmínky
  - celočíselné programování
  - meta-heuristiky (lokální prohledávání)

Datové přenosy: cvičení

Pro zadanou počítačovou síť uveďte uveďte seznam proměnných. Navrhněte, jak by mohl být datový přednos realizován a uveďte odpovídající hodnoty proměnných reprezentující řešení optimalizující celkovou latenci. Jaká je celková latence?

Ohodnocení linku l

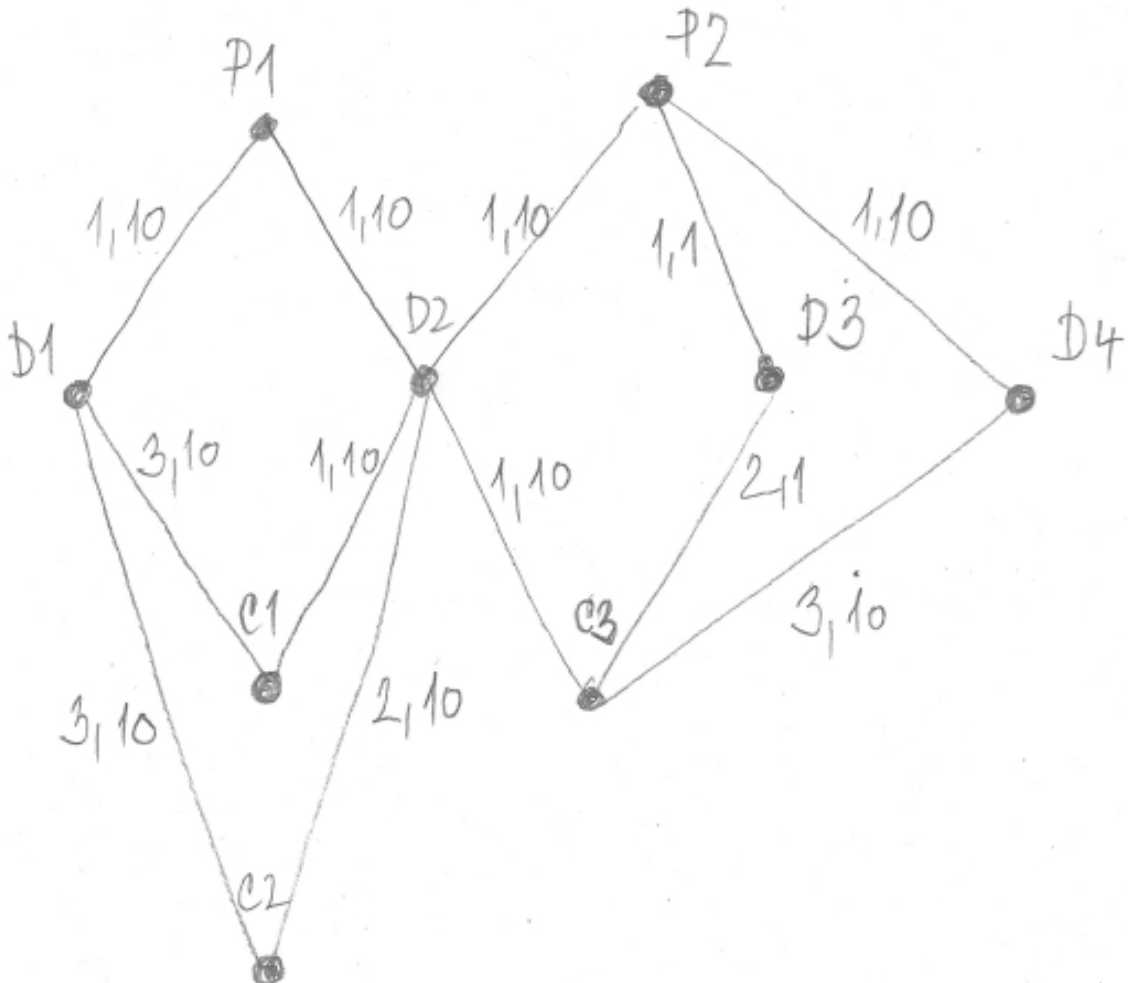
- $latency(l), capacity(l)$

Stream $s1$

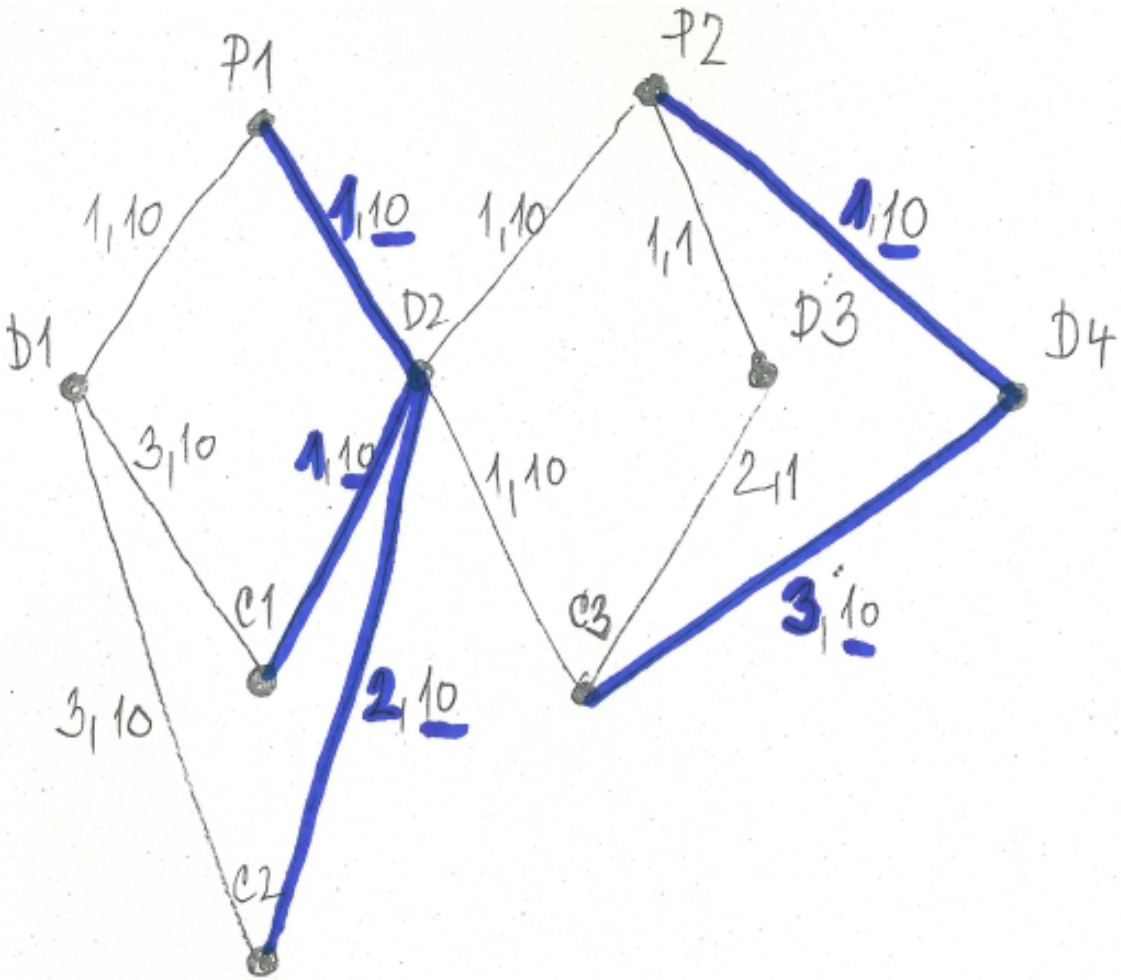
- $bandwith(s1) = 5$
- $producer(s1) = P1$
- $consumers(s1) = \{C1, C2\}$

Stream $s2$

- $bandwith(s2) = 5$
- $producer(s2) = P2$
- $consumers(s2) = \{C3\}$



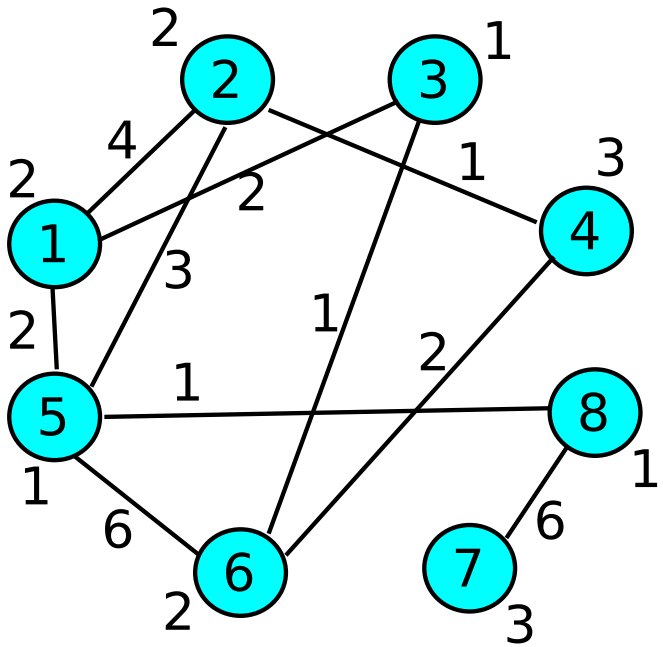
Datové přednosy: výsledný graf



Paralelní komunikující úlohy

Paralelní aplikace

- n komunikujících úloh
- m procesorů
- několik úloh prováděno zároveň na každém procesoru
- tj. opět plánování pro daný časový okamžik
- kromě přenosů explicitně uvažována zátěž na uzlech



Grafová reprezentace

- hranově a vrcholově ohodnocený neorientovaný graf
- ohodnocené vrcholy: úlohy s danou výpočetní náročností
- ohodnocené hrany: průběžně komunikující úlohy s komunikační náročností

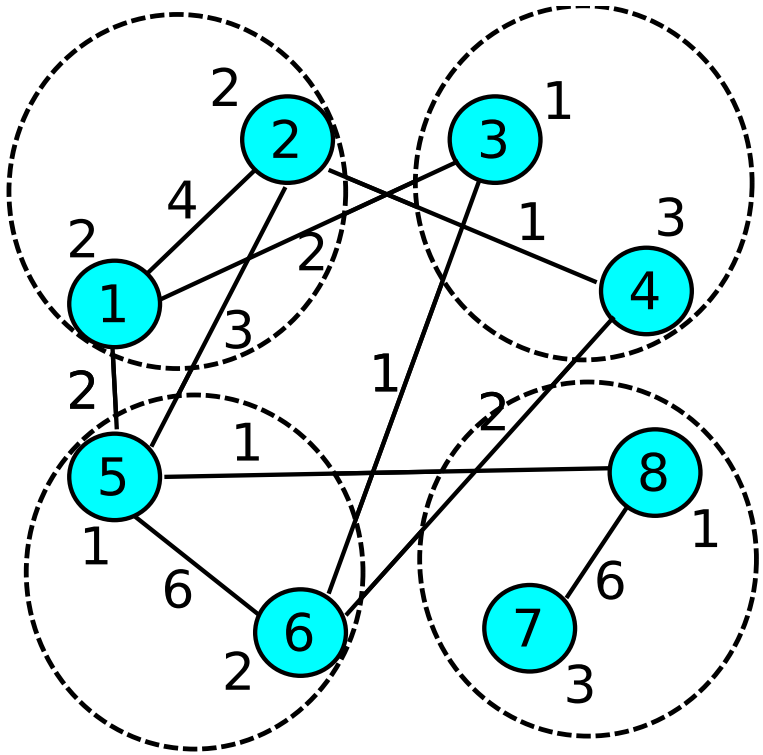
Paralelní komunikující úlohy

Paralelní aplikace

- n komunikujících úloh
- m procesorů
- několik úloh prováděno zároveň na každém procesoru
- tj. opět plánování pro daný časový okamžik
- kromě přenosů explicitně uvažována zátěž na uzlech

Grafová reprezentace

- hranově a vrcholově ohodnocený neorientovaný graf
- ohodnocené vrcholy: úlohy s danou výpočetní náročností
- ohodnocené hrany: průběžně komunikující úlohy s komunikační náročností



Vyvažování zátěže *(load balancing)*:
přiřazení úloh na procesory tak, aby

- byla vyvážená zátěž jednotlivých procesorů
- byla minimalizována komunikace úloh na různých procesorech

Rozdělení grafu

Formulace problému vyvažování zátěže jako
problému rozdělení grafu *(graph partitioning)*

Rozdělení grafu $G = (V, E)$ na $V = V_1 \cup \cdots \cup V_m$ tak, že je

- $V_1 \cap \cdots \cap V_m = \emptyset$
- $G_1 = (V_1, E_1), \ldots, G_m = (V_m, E_m)$
- E_i tvořeno hranami, jejichž oba vrcholy patří do V_i
- součet ohodnocení vrcholů v jednotlivých V_i „zhruba stejný"
- součet ohodnocení hran $E \backslash \{E_1 \cup \cdots E_m\}$ spojujících různé V_j a V_k minimalizován

Rozdělení grafu a bisekce grafu

Speciální případ: $V = V_1 \cup V_2$ bisekce grafu (graph bisection), tj. z grafu $G = (V, E)$ vytvoříme dva podgrafy (V_1, E_1) (V_2, E_2) tak, že

- $V = V_1 \cup V_2$, $V_1 \cap V_2 = \emptyset$
- E_i tvořeno hranami, jejichž oba vrcholy patří do V_i, tj.

  $E_1, E_2 \subset E$, $E_1 \cap E_2 = \emptyset$, E_i
- součet ohodnocení vrcholů ve V_1 a V_2 je „zhruba stejný"
- součet ohodnocení hran $E \backslash \{E_1 \cup E_2\}$ spojující vrcholy z V_1 a V_2 je minimalizován

Jak nalézt vhodné rozdělení grafu?

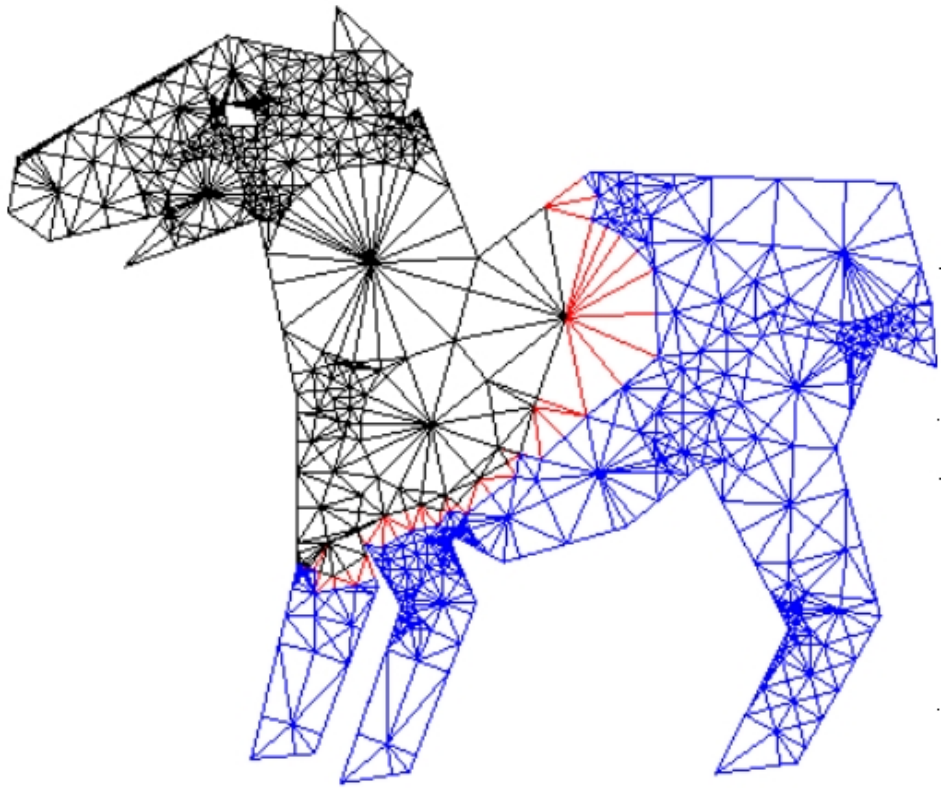
- problém optimálního rozdělení je NP-úplný
  - už pro bisekci: prohledání všech podmnožin množiny vrcholů (podmnožina a její doplněk tvoří V_1 a V_2)
- nutné použít dobré heuristiky

Heuristika: opakovaná bisekce grafu

Základní používaný princip při rozdělení grafu:

rozdělení množiny vrcholů V na 2^k částí:

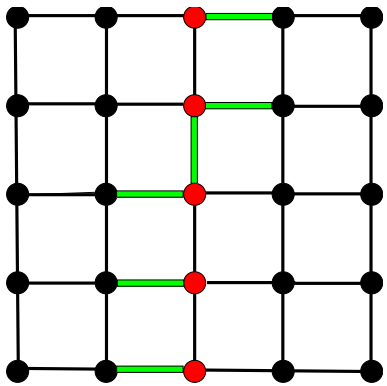
rekurzivní bisekce grafu k-krát



58 dělících hran

Dělící hrany (edge separator) vs. dělící vrcholy (vertex separator)

- Dělící hrany: $E_S \subseteq E$ dělí G po odstranění E_S z E na dvě stejně velké nesouvisející komponenty V: V_1 a V_2
- Dělící vrcholy: $V_S \subseteq V$ dělí G po odstranění V_S a všech jejich hran na dvě stejně velké nesouvisející komponenty V: V_1 a V_2



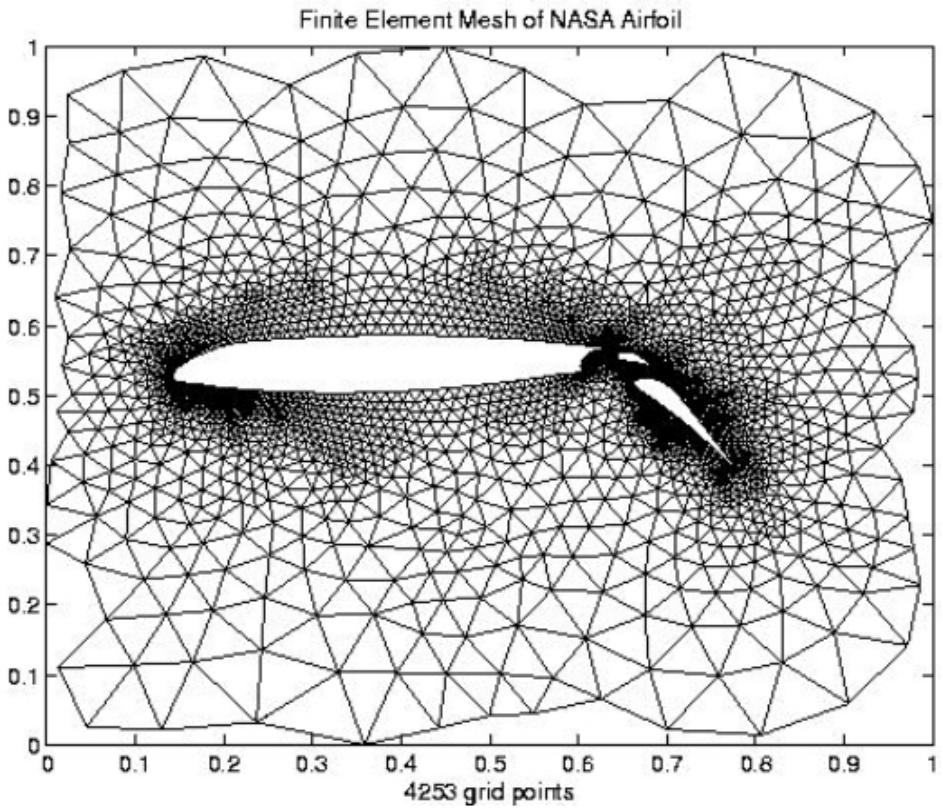
E_S = zelené hrany

V_S = červené vrcholy

Rozdělení se souřadnicemi vrcholů (partitioning with nodal coordinates)

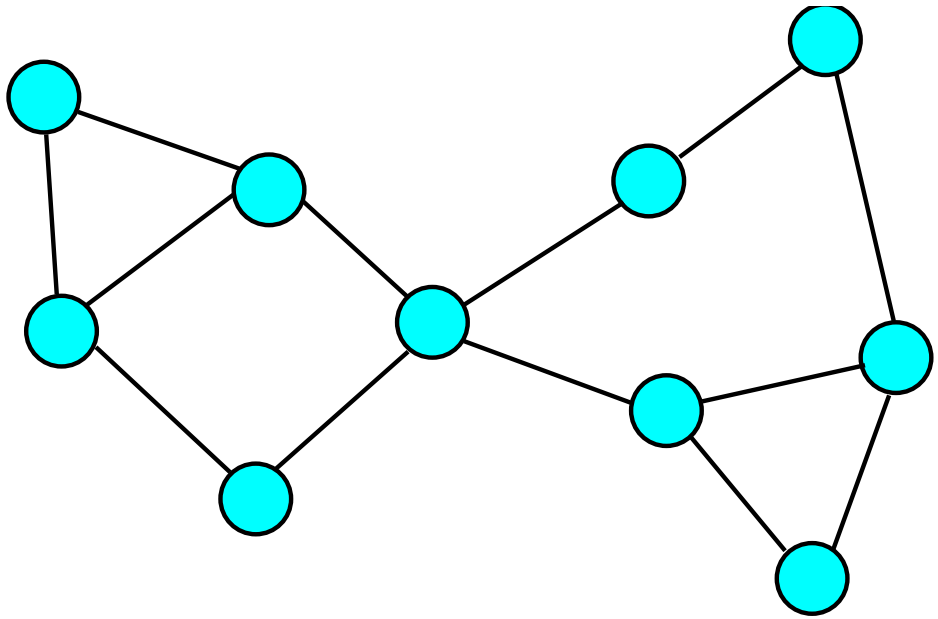
Myšlenka rozdělení pomocí souřadnic vrcholů

- každý vrchol má souřadnice v prostoru → rozdělení prostoru

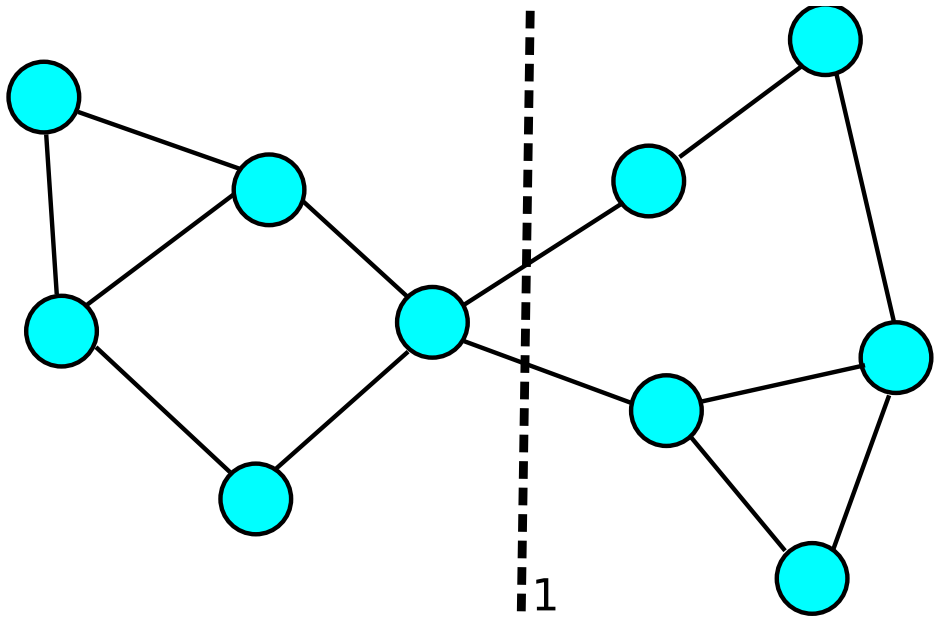


- pomocí dělicí přímky, která dělí vrcholy v prostoru na poloviny

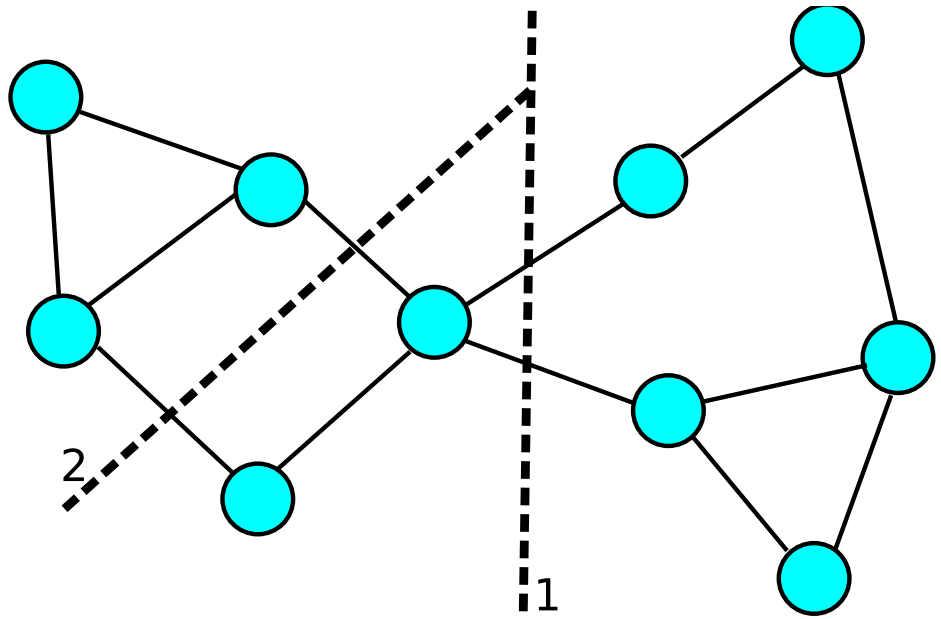
Opakovaná bisekce grafu s dělící přímkou: příklad



Opakovaná bisekce grafu s dělící přímkou: příklad



Opakovaná bisekce grafu s dělící přímkou: příklad



Opakovaná bisekce grafu s dělící přímkou: příklad