Static Load Balancing Applied to Time Dependent Mechanical Problems

O. Medek[1], J. Kruis[2], Z. Bittnar[2], P. Tvrdík[1]

[1]Katedra počítačů
České vysoké učení technické, Praha

[2]Katedra stavební mechaniky
České vysoké učení technické, Praha

Seminář numerické analýzy 2005

Obsah

Obsah

Časově závislé mechanické problémy

- Závislost na čase.

Časově závislé mechanické problémy

- Závislost na čase.
- Zanedbatelné inerciální síly.

Časově závislé mechanické problémy

- Závislost na čase.
- Zanedbatelné inerciální síly.
- Př.: analýza reaktorové tlakové nádoby v jaderných elektrárnách.

Řešení časově závislý mechanických problémů

- Prostorová diskretizace: metoda konečných prvků (MKP).

Řešení časově závislý mechanických problémů

- Prostorová diskretizace: metoda konečných prvků (MKP).
- Časová diskretizace: metoda konečných diferencí + linearizace (Newtonov-Raphsonovou metodou) $\Rightarrow$ iterace.

Řešení časově závislý mechanických problémů

- Prostorová diskretizace: metoda konečných prvků (MKP).
- Časová diskretizace: metoda konečných diferencí + linearizace (Newtonov-Raphsonovou metodou) $\Rightarrow$ iterace.
- V každé iteraci se řeší systém lineárních rovnic (SLR) $\boldsymbol{Ax} = \boldsymbol{b}$.

Řešení časově závislý mechanických problémů

- Prostorová diskretizace: metoda konečných prvků (MKP).
- Časová diskretizace: metoda konečných diferencí + linearizace (Newtonov-Raphsonovou metodou) $\Rightarrow$ iterace.
- V každé iteraci se řeší systém lineárních rovnic (SLR) $\boldsymbol{Ax} = \boldsymbol{b}$.
- Tyto SLR mají shodnou strukturu.

Řešení časově závislý mechanických problémů

- Prostorová diskretizace: metoda konečných prvků (MKP).
- Časová diskretizace: metoda konečných diferencí + linearizace (Newtonov-Raphsonovou metodou) $\Rightarrow$ iterace.
- V každé iteraci se řeší systém lineárních rovnic (SLR) $\boldsymbol{Ax} = \boldsymbol{b}$.
- Tyto SLR mají shodnou strukturu.
- Dále předpokládáme úlohy se symetrickými, pozitivně definitními řídkými a velkými SLR.

Obsah

Paralelizace řešení systému lineárních rovnic

Metoda Schurových doplňků

1. Doménová dekompozice (DD).

Paralelizace řešení systému lineárních rovnic

Metoda Schurových doplňků

1. Doménová dekompozice (DD).
2. Přečíslování proměnných.

Paralelizace řešení systému lineárních rovnic

Metoda Schurových doplňků

1. Doménová dekompozice (DD).
2. Přečíslování proměnných.

---

3. Sestavení podmatic.

Paralelizace řešení systému lineárních rovnic

Metoda Schurových doplňků

1. Doménová dekompozice (DD).
2. Přečíslování proměnných.

---

3. Sestavení podmatic.
4. Částečná faktorizace podmatic (výpočet Schurových doplňků).

Paralelizace řešení systému lineárních rovnic

Metoda Schurových doplňků

1. Doménová dekompozice (DD).
2. Přečíslování proměnných.

---

3. Sestavení podmatic.
4. Částečná faktorizace podmatic (výpočet Schurových doplňků).
5. Řešení redukovaného systému.

Paralelizace řešení systému lineárních rovnic

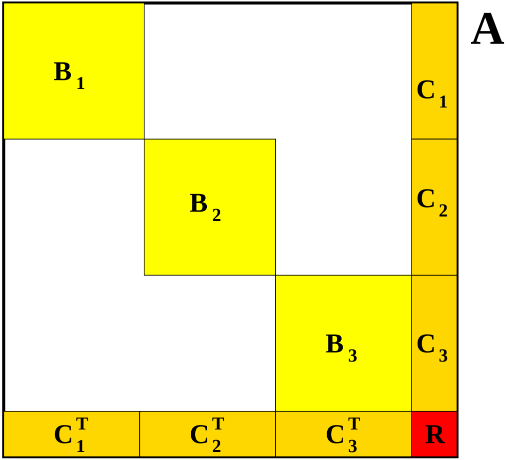
Metoda Schurových doplňků

1. Doménová dekompozice (DD).
2. Přečíslování proměnných.

---

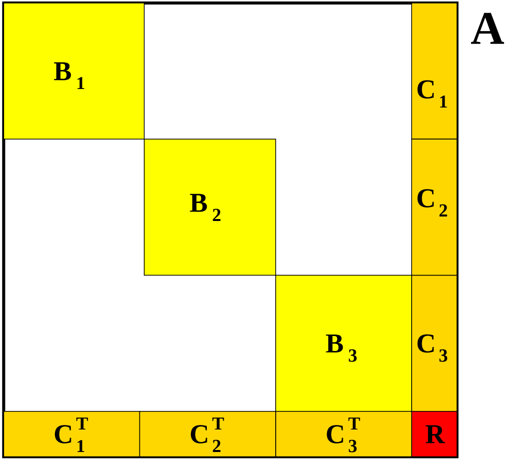
3. Sestavení podmatic.
4. Částečná faktorizace podmatic (výpočet Schurových doplňků).
5. Řešení redukovaného systému.
6. Zpětná substituce na podmaticích.

Doménová dekompozice (DD)



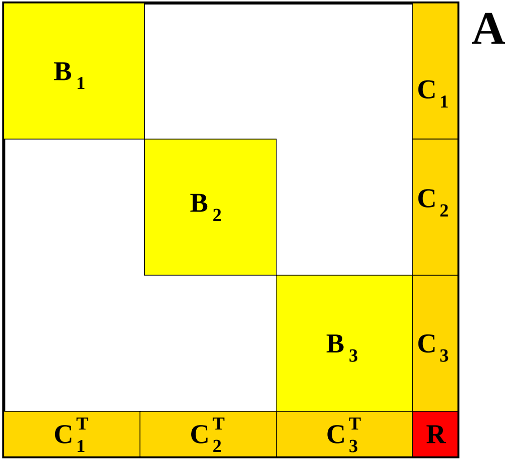
- Blokově–šípový tvar.

Doménová dekompozice (DD)



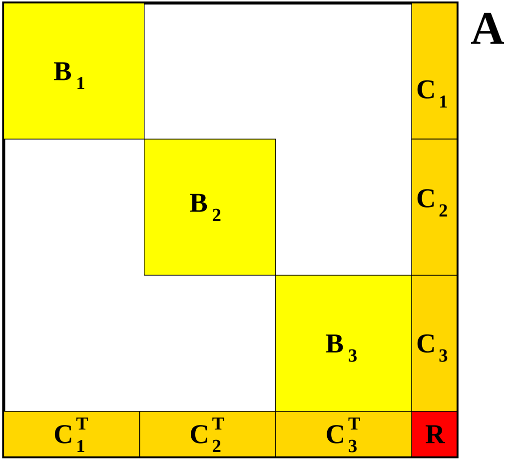
- Blokově–šípový tvar.
- ***R*** je matice redukovaného problému.

Doménová dekompozice (DD)



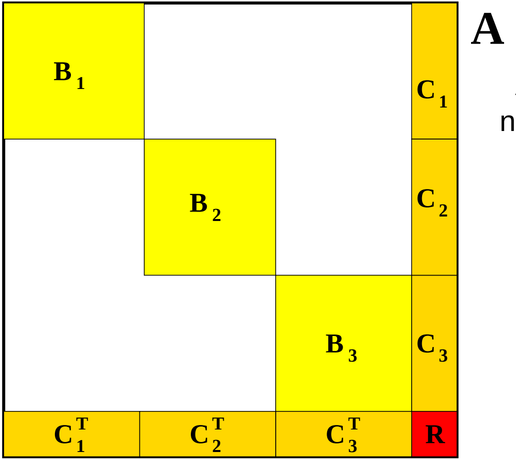
- Blokově–šípový tvar.
- ***R*** je matice redukovaného problému.
- Řád vnitřních $\boldsymbol{B_k}$ zhruba stejný.

Doménová dekompozice (DD)

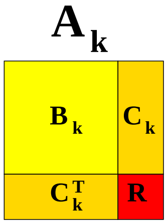


- Blokově–šípový tvar.
- $\boldsymbol{R}$ je matice redukovaného problému.
- Řád vnitřních $\boldsymbol{B}_k$ zhruba stejný.
- Šířka hraničních $\boldsymbol{C}_k$ a $\boldsymbol{R}$ je minimalizována.

Doménová dekompozice (DD)



A_k je podmatice vytvořená na procesoru k.



Zápis blokově–šípový tvaru SLR

$$
\begin{aligned}
B_1 x_1 + C_1 x_R &= b_1 \\
B_2 x_2 + C_2 x_R &= b_3 \\
B_3 x_3 + C_3 x_R &= b_3 \\
C_1^T x_1 + C_1^T x_2 + C_1^T x_3 + R x_R &= b_R
\end{aligned}
$$

Zápis blokově–šípový tvaru SLR

$$
\begin{aligned}
B_1 x_1 + C_1 x_R &= b_1 \\
B_2 x_2 + C_2 x_R &= b_3 \\
B_3 x_3 + C_3 x_R &= b_3 \\
C_1^T x_1 + C_1^T x_2 + C_1^T x_3 + R x_R &= b_R
\end{aligned}
$$

x_k vnitřní proměnné (x_1, x_2, x_3).

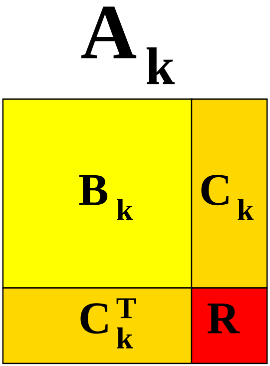
Zápis blokově–šípový tvaru SLR

$$
\begin{aligned}
B_1 x_1 + C_1 x_R &= b_1 \\
B_2 x_2 + C_2 x_R &= b_3 \\
B_3 x_3 + C_3 x_R &= b_3 \\
C_1^T x_1 + C_1^T x_2 + C_1^T x_3 + R x_R &= b_R
\end{aligned}
$$

x_k vnitřní proměnné (x_1, x_2, x_3).

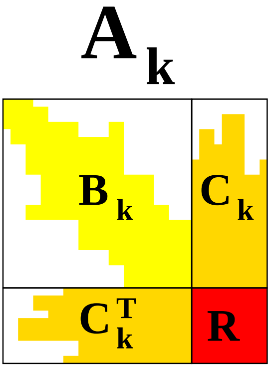
x_R hraniční proměnné.

Částečná faktorizace podmatic



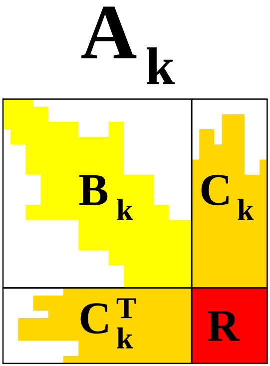
- Eliminují se pouze vnitřní proměnné x_k.

Částečná faktorizace podmatic



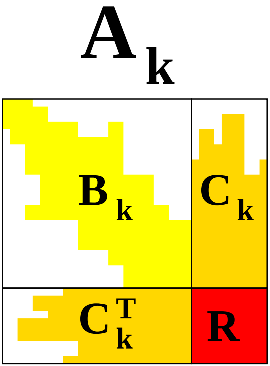
- Eliminují se pouze vnitřní proměnné $\boldsymbol{x_k}$.
- Podmatice $\boldsymbol{A_k}$ jsou řídké.

Částečná faktorizace podmatic



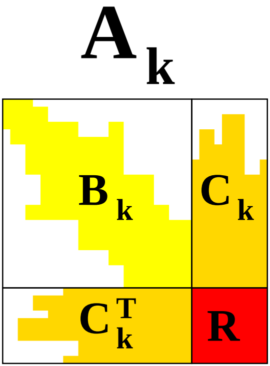
- Eliminují se pouze vnitřní proměnné $\boldsymbol{x_k}$.
- Podmatice $\boldsymbol{A_k}$ jsou řídké.
- Částečná faktorizace se provádí obálkovou (skyline) metodou.

Obálková metoda



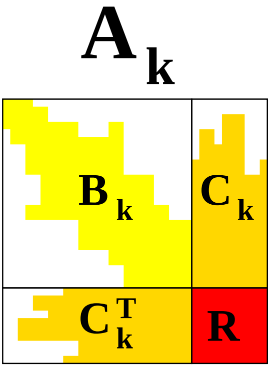
- Obálka = “množina prvků okolo hlavní diagonály”. Mimo obálku jsou jen 0.

Obálková metoda



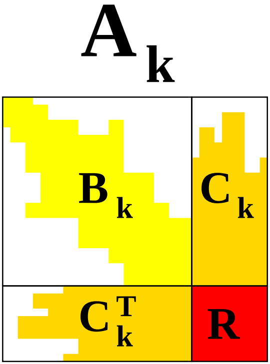
- Obálka = “množina prvků okolo hlavní diagonály”. Mimo obálku jsou jen 0.
- Částečná faktorizace se provádí pouze uvnitř obálky.

Obálková metoda



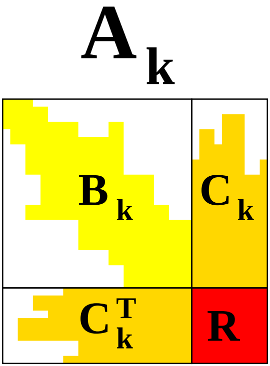
- Obálka = “množina prvků okolo hlavní diagonály”. Mimo obálku jsou jen 0.
- Částečná faktorizace se provádí pouze uvnitř obálky.
- V paměti jsou pouze prvky uvnitř obálky.

Obálková metoda



- Obálka = “množina prvků okolo hlavní diagonály”. Mimo obálku jsou jen 0.
- Částečná faktorizace se provádí pouze uvnitř obálky.
- V paměti jsou pouze prvky uvnitř obálky.
- Přečíslování proměnných $\longrightarrow$ minimalizace obálky.

Obálková metoda



- Obálka = “množina prvků okolo hlavní diagonály”. Mimo obálku jsou jen 0.
- Částečná faktorizace se provádí pouze uvnitř obálky.
- V paměti jsou pouze prvky uvnitř obálky.
- Přečíslování proměnných $\longrightarrow$ minimalizace obálky.
- Přečíslování: hraniční Sloanův algoritmus.

Příklad obálky

1	●								
2	●	●							
3	●	●	●						
4	0	●	●	●					
5	0	●	0	0	●				
6	0	0	●	0	●	●			
7	●	0	●	0	0	●	●		
8	●	0	●	●	0	0	●	●	
9	0	0	0	●	0	0	0	●	●

●: diagonální prvek.
0: nula mimo obálku.
● 0 : nuly a nenulové prvky uvnitř obálky.
1 – 6: vnitřní proměnné.
7 – 9: hraniční proměnné.

Obsah

Problém vyváženosti výpočtu

- DD dělí na podmatice $\boldsymbol{A_k}$ zhruba stejného řádu.

Problém vyváženosti výpočtu

- DD dělí na podmatice $\boldsymbol{A_k}$ zhruba stejného řádu.
- Doba částečné faktorizace závisí na velikosti obálky.
  V praxi se doby částečných faktorizací podmatic mohou lišit (až 2,5–krát).

Problém vyváženosti výpočtu

- DD dělí na podmatice $\boldsymbol{A_k}$ zhruba stejného řádu.
- Doba částečné faktorizace závisí na velikosti obálky.
  V praxi se doby částečných faktorizací podmatic mohou lišit (až 2,5–krát).
- Navíc přečíslování mění (menšuje) velikost obálky až po DD.

Problém vyváženosti výpočtu

- DD dělí na podmatice $\boldsymbol{A_k}$ zhruba stejného řádu.
- Doba částečné faktorizace závisí na velikosti obálky. V praxi se doby částečných faktorizací podmatic mohou lišit (až 2,5–krát).
- Navíc přečíslování mění (menšuje) velikost obálky až po DD.

JAK UDĚLAT VÝPOČETNĚ VYVÁŽENOU DEKOMPOZICI?

Obsah

Spojení DD a přečíslování

1. Doménová dekompozice
2. Přečíslování

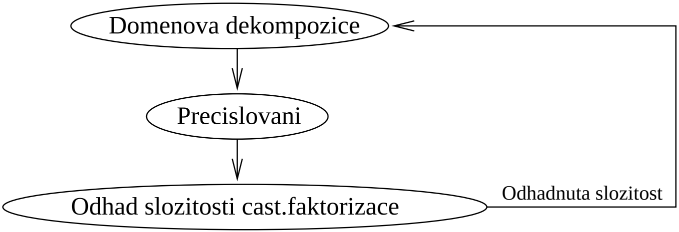
Spojení DD a přečíslování

1. Doménová dekompozice
2. Přečíslování

Spojení DD a přečíslování

1. Doménová dekompozice
2. Přečíslování

Quality Balancing heuristika



Quality Balancing heuristika

- Quality = míra, která se vyvažuje:
  - paměťové nároky,
  - výpočetní složitost.

Quality Balancing heuristika

- Quality = míra, která se vyvažuje:
  - paměťové nároky,
  - výpočetní složitost.
- Prezentováno na EUROPAR'04 a PDCN'05 na jednoduchých úlohách mechaniky.

Quality Balancing heuristika

- Quality = míra, která se vyvažuje:
  - paměťové nároky,
  - výpočetní složitost.
- Prezentováno na EUROPAR'04 a PDCN'05 na jednoduchých úlohách mechaniky.
- Vyvážený výpočet trvá kratší dobu.

Quality Balancing heuristika

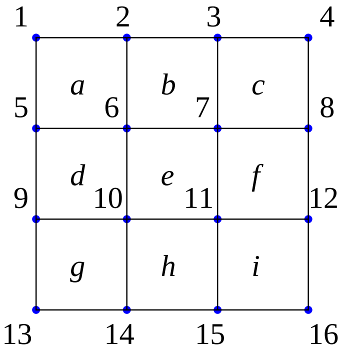
- Quality = míra, která se vyvažuje:
  - paměťové nároky,
  - výpočetní složitost.
- Prezentováno na EUROPAR'04 a PDCN'05 na jednoduchých úlohách mechaniky.
- Vyvážený výpočet trvá kratší dobu.
- Ale QB je značně pomalejší nežli klasická DD.

Quality Balancing heuristika

- Quality = míra, která se vyvažuje:
  - paměťové nároky,
  - výpočetní složitost.
- Prezentováno na EUROPAR'04 a PDCN'05 na jednoduchých úlohách mechaniky.
- Vyvážený výpočet trvá kratší dobu.
- Ale QB je značně pomalejší nežli klasická DD.
- Tento příspěvek je zaměřen na vyvažování výpočetní složitosti řešení časově závislých mechanických problémů, kde se projeví výhody QB.

Obsah

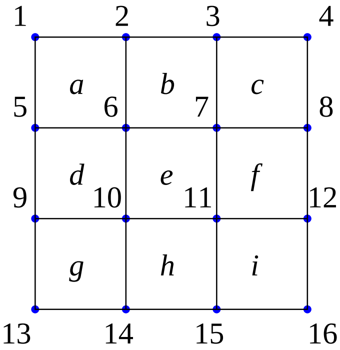
Síť konečných prvků (KP)

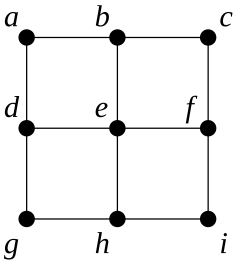


Elementy (konečné prvky) $a \dots i$. Uzly $1 \dots 16$.
Každý uzel obsahuje stupně volnosti (SV, proměnné).

Duální graf

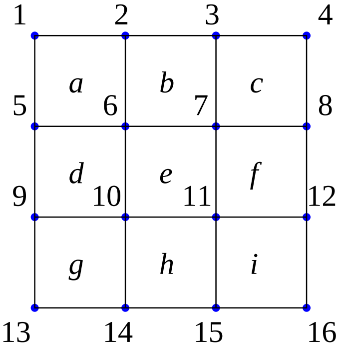
Síť KP reprezentovaná duálním grafem G^{D}.

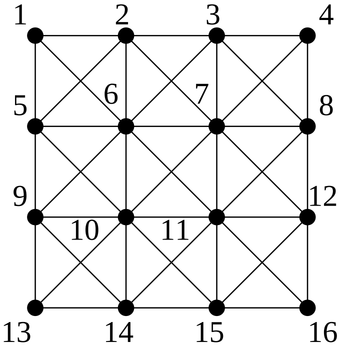




Uzlový graf

Síť KP reprezentovaná uzlovým grafem G^{N}.





DD dělením grafu

DD se provádí pomocí dělení G^{D} *hranovým řezem* (víceúrovňový dělič METIS).

DD dělením grafu

DD se provádí pomocí dělení G^{D} *hranovým řezem* (víceúrovňový dělič METIS).

$\Rightarrow$ Rozdělení sítě KP na podsítě KP.

DD dělením grafu

DD se provádí pomocí dělení G^{D} *hranovým řezem* (víceúrovňový dělič METIS).

$\Rightarrow$ Rozdělení sítě KP na podsítě KP.

$\Rightarrow$ Rozdělení G^{N} *vrcholovým řezem*.

DD dělením grafu

DD se provádí pomocí dělení G^{D} *hranovým řezem* (víceúrovňový dělič METIS).

$\Rightarrow$ Rozdělení sítě KP na podsítě KP.

$\Rightarrow$ Rozdělení G^{N} *vrcholovým řezem*.

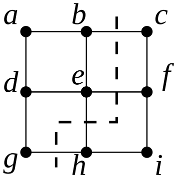
$\Rightarrow$ Dekompozice $\boldsymbol{A}$ na podmatice $\boldsymbol{A_k}$.

DD dělením grafu

DD se provádí pomocí dělení G^{D} *hranovým řezem* (víceúrovňový dělič METIS).

- ⇒ Rozdělení sítě KP na podsítě KP.
- ⇒ Rozdělení G^{N} *vrcholovým řezem*.
- ⇒ Dekompozice $\boldsymbol{A}$ na podmatice $\boldsymbol{A_k}$.
- ⇒ Uzly (proměnné) patřící k více než jedné podsíti (podmatici) jsou *hraniční*; ostatní jsou *vnitřní*.

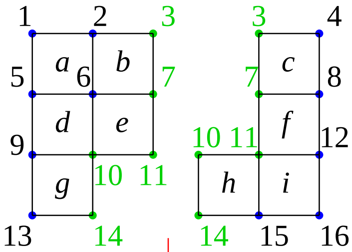
DD dělením grafu (obrázek)



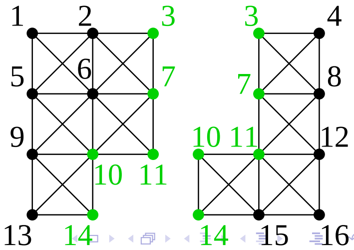
Rozdeleny G^D

Podsite KP

- vnitrni uzly
- hranicni uzly

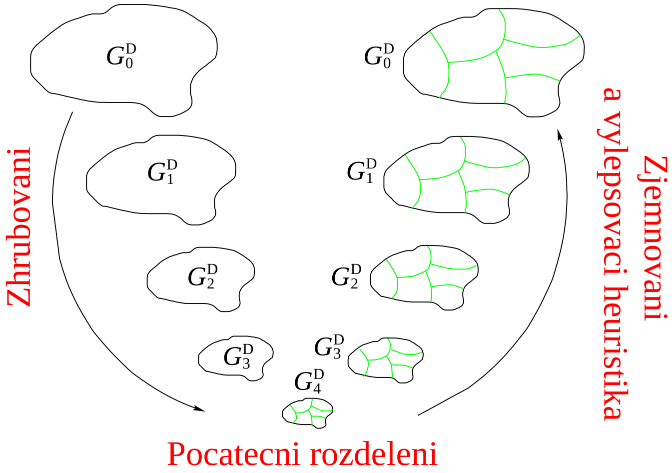


Rozdeleny G^N



- vnitrni vrcholy
- hranicni vrcholy

Víceúrovňové dělení grafu



Vylepšovací heuristika

- Přesouvá supervrcholy (množiny vrcholů) mezi podgrafy.

Vylepšovací heuristika

- Přesouvá supervrcholy (množiny vrcholů) mezi podgrafy.
- Vyvažuje počet vrcholů v podgrafech a zmenšuje hranový řez.

Vylepšovací heuristika

- Přesouvá supervrcholy (množiny vrcholů) mezi podgrafy.
- Vyvažuje počet vrcholů v podgrafech a zmenšuje hranový řez.
- Nejvíce ze všech fází ovlivňuje výsledné dělení.

Obsah

QB heuristika

- Rozšiřuje a modifikuje vylepšovací heuristiku.

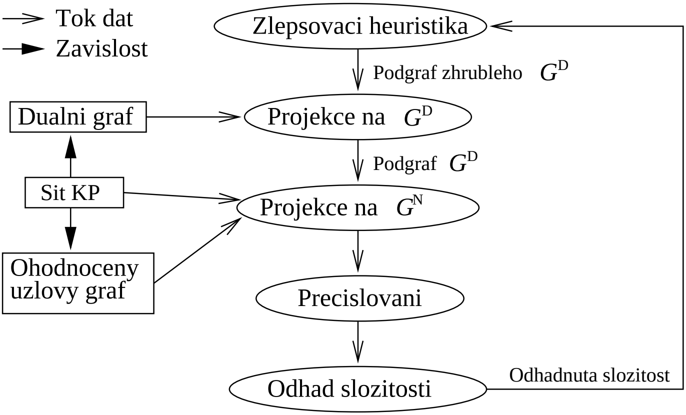
QB heuristika

- Rozšiřuje a modifikuje vylepšovací heuristiku.
- Pro každý podgraf spočte odhad výpočetní zátěže (#FLOPs) částečné faktorizace příslušné podmatice.

QB heuristika

- Rozšiřuje a modifikuje vylepšovací heuristiku.
- Pro každý podgraf spočte odhad výpočetní zátěže (#FLOPs) částečné faktorizace příslušné podmatice.
- Vyvažuje odhady výpočetních zátěží a zmenšuje hranový řez.

QB heuristika (obrázek)



Obsah

Testování

- Testováno na výpočtech stárnutí reaktorové nádoby.

Testování

- Testováno na výpočtech stárnutí reaktorové nádoby.
- Pouze pro 130 časových kroků. (V praxi je jich potřeba 13000).

Testování

- Testováno na výpočtech stárnutí reaktorové nádoby.
- Pouze pro 130 časových kroků. (V praxi je jich potřeba 13000).
- Dekompozice na 4, 6, 8 a 10 domén, nejprve METISem, pak QB.

Testování

- Testováno na výpočtech stárnutí reaktorové nádoby.
- Pouze pro 130 časových kroků. (V praxi je jich potřeba 13000).
- Dekompozice na 4, 6, 8 a 10 domén, nejprve METISem, pak QB.
- Testováno na Linuxovém clusteru. Každý stroj: Pentium 4, 3,2 GHz, 3GB paměti.

Testování

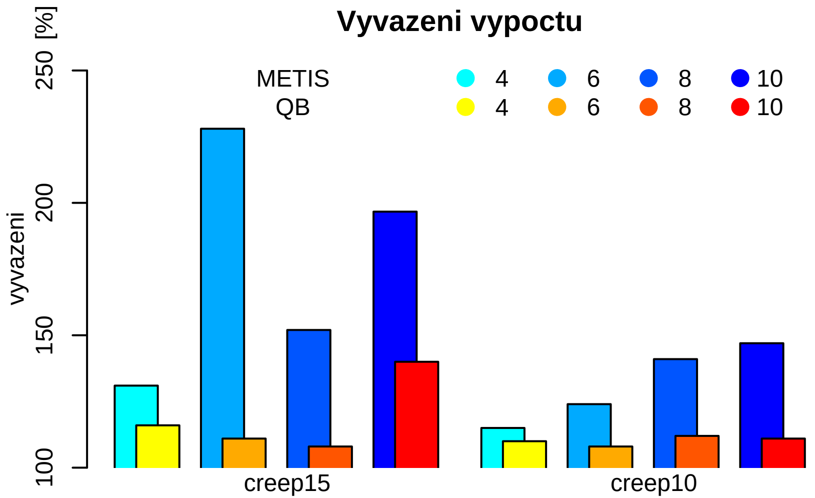
- Testováno na výpočtech stárnutí reaktorové nádoby.
- Pouze pro 130 časových kroků. (V praxi je jich potřeba 13000).
- Dekompozice na 4, 6, 8 a 10 domén, nejprve METISem, pak QB.
- Testováno na Linuxovém clusteru. Každý stroj: Pentium 4, 3,2 GHz, 3GB paměti.
- Řešič SIFEL `http://cml.fsv.cvut.cz/~sifel/`, zkompilován `gcc` s optimalizací `-O3`.

Popis testovacích problémů

- Diskretizace reaktorové nádoby pomocí čtyřstěnů.
- 2 Sítě KP různé jemnosti: creep15 a creep10.

	creep15	creep10
#elementů	47090	152464
#uzlů	13529	38235
#SV	3	3

Vyvážení výpočtu



Výsledné zrychlení

T_{M} (T_{QB}) doba 130 časových kroků [sec] po dělení METISem (QB).

Δ úspora času výpočtu po dělení QB [%].

t_{QB} doba běhu QB [sec].

	creep15				creep10			
#domén	T_{M}	T_{QB}	Δ	t_{QB}	T_{M}	T_{QB}	Δ	t_{QB}
4	2432	2294	6	10	21256	20743	2	56
6	2543	1097	57	6	11607	10264	12	49
8	847	711	16	7	7710	6255	19	49
10	844	567	33	7	5550	4084	26	46

Obsah

Závěr

- QB heuristika:
  a) vyvažuje paralelní výpočet,

Závěr

- QB heuristika:
  a) vyvažuje paralelní výpočet,
  b) zkracuje dobu řešení,

Závěr

- QB heuristika:
  a) vyvažuje paralelní výpočet,
  b) zkracuje dobu řešení,
  c) prodlužuje dobu dekompozice,

Závěr

- QB heuristika:
  a) vyvažuje paralelní výpočet,
  b) zkracuje dobu řešení,
  c) prodlužuje dobu dekompozice,

$\Rightarrow$ je vhodná pro dekompozici časově závislých mechanických problémů.

Závěr

- QB heuristika:
  a) vyvažuje paralelní výpočet,
  b) zkracuje dobu řešení,
  c) prodlužuje dobu dekompozice,

$\Rightarrow$ je vhodná pro dekompozici časově závislých mechanických problémů.

- Další výzkum
  - testování na dalších úlohách.

Závěr

- QB heuristika:
  a) vyvažuje paralelní výpočet,
  b) zkracuje dobu řešení,
  c) prodlužuje dobu dekompozice,

$\Rightarrow$ je vhodná pro dekompozici časově závislých mechanických problémů.

- Další výzkum
  - testování na dalších úlohách.
  - testování na větším počtu procesorů.