Anotace

- Středník II (alias Checkpoint Bravo)!! 6. 5. 2010
- programování her,
- rozděl a panuj (poznámky).

Teorie her

- Kombinatorická hra je hrou dvou hráčů. Stav hry je určen pozicí nějakých předmětů. Všechny zúčastněné předměty jsou viditelné. Jde o tzv. hru s úplnou informací.

Teorie her

- Kombinatorická hra je hrou dvou hráčů. Stav hry je určen pozicí nějakých předmětů. Všechny zúčastněné předměty jsou viditelné. Jde o tzv. hru s úplnou informací.
- Příklad: Nimm, Podivná hra, Dáma, Šachy, Halma, Mlýn, Otrávená čokoláda...

Teorie her

- Kombinatorická hra je hrou dvou hráčů. Stav hry je určen pozicí nějakých předmětů. Všechny zúčastněné předměty jsou viditelné. Jde o tzv. hru s úplnou informací.
- Příklad: Nimm, Podivná hra, Dáma, Šachy, Halma, Mlýn, Otrávená čokoláda...
- Kombinatorickými hrami nejsou: Poker, Prší, Mariáš, Black Jack, závody formulí...

Teorie her

- Kombinatorická hra je hrou dvou hráčů. Stav hry je určen pozicí nějakých předmětů. Všechny zúčastněné předměty jsou viditelné. Jde o tzv. hru s úplnou informací.
- Příklad: Nimm, Podivná hra, Dáma, Šachy, Halma, Mlýn, Otrávená čokoláda...
- Kombinatorickými hrami nejsou: Poker, Prší, Mariáš, Black Jack, závody formulí...
- Zaměříme se na hrací část, ne na vstup a výstup.

Teorie her

- Kombinatorická hra je hrou dvou hráčů. Stav hry je určen pozicí nějakých předmětů. Všechny zúčastněné předměty jsou viditelné. Jde o tzv. hru s úplnou informací.
- Příklad: Nimm, Podivná hra, Dáma, Šachy, Halma, Mlýn, Otrávená čokoláda...
- Kombinatorickými hrami nejsou: Poker, Prší, Mariáš, Black Jack, závody formulí...
- Zaměříme se na hrací část, ne na vstup a výstup.
- U her předpokládáme, že hrají rozumně se chovající jedinci (s motivací vyhrát).

Shannonova věta

Theorem (Shannon)

Každá kombinatorická hra má pro některého z hráčů neprohrávající strategii.

Důkaz.

Náznak: Buďto platí, že si jeden z hráčů může vynutit zacyklení hry (a tak neprohrát), nebo budeme zkoumat predikáty:
Existuje náš tah, že pro každý tah protihráče existuje náš tah, že pro každý tah protihráče... protihráč prohraje.
Pro každý náš tah existuje tah protihráče, že pro každý náš tah... my prohrajeme.
Formule jsou konečné, počty tahů jsou také konečné, jsou to vzájemně negace a lze je algoritmicky rozhodnout. □

Shannonova věta

Theorem (Shannon)

Každá kombinatorická hra má pro některého z hráčů neprohrávající strategii.

Corollary

Pokud je ve hře remíza vyloučena, jeden z hráčů má vyhrávající strategii.

Graf hry

- Ke hře (případně její instanci) definujeme orientovaný graf:
- Vrcholy: Stavy hry,
- Hrany: Možnosti přechodů mezi jednotlivými stavy.
- Příklad pro Nimm, kdy odebíráme 1 nebo 2 sirky (na tabuli).
- Každému stavu můžeme přiřadit barvu říkající, zda se odtud vyhrává nebo prohrává.

Příklad grafů her

- Na hracím plánu tvaru orientovaného grafu vyrážíme z určeného vrcholu. Taháme jedním padesátníkem.

Příklad grafů her

- Na hracím plánu tvaru orientovaného grafu vyrážíme z určeného vrcholu. Taháme jedním padesátníkem.
- Máme dojet do jednoho z cílových vrcholů. Kdo dojede, vyhraje.

Příklad grafů her

- Na hracím plánu tvaru orientovaného grafu vyrážíme z určeného vrcholu. Taháme jedním padesátníkem.
- Máme dojet do jednoho z cílových vrcholů. Kdo dojede, vyhraje.
- Graf hry máme přímo zadaný a jde jen o to, který vrchol vyhrává.

Příklad grafů her

- Na hracím plánu tvaru orientovaného grafu vyrážíme z určeného vrcholu. Taháme jedním padesátníkem.
- Máme dojet do jednoho z cílových vrcholů. Kdo dojede, vyhraje.
- Graf hry máme přímo zadaný a jde jen o to, který vrchol vyhrává.
- Podivná hra: Graf hry si zakreslíme na šachovnici. Vrcholy jsou políčka, hrany vedou tudy, kudy může figurka.

Příklad grafů her

- Na hracím plánu tvaru orientovaného grafu vyrážíme z určeného vrcholu. Taháme jedním padesátníkem.
- Máme dojet do jednoho z cílových vrcholů. Kdo dojede, vyhraje.
- Graf hry máme přímo zadaný a jde jen o to, který vrchol vyhrává.
- Podivná hra: Graf hry si zakreslíme na šachovnici. Vrcholy jsou políčka, hrany vedou tudy, kudy může figurka.
- Stačí říct, ze kterého vrcholu se vyhrává, prohrává, nebo zda existuje cyklus, po kterém mají oba hráči zájem bloudit (resp. zda si někdo z hráčů může bloudění po této kružnici vynutit).

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.
- V tomto stromě nás zajímá, zda existuje větev, po které pokud pojedeme, tak vyhrajeme.

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.
- V tomto stromě nás zajímá, zda existuje větev, po které pokud pojedeme, tak vyhrajeme.
- Tato větev se pozná tak, že ve všech synech jejího koncového vrcholu existuje vyhrávající cesta, tedy ...

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.
- V tomto stromě nás zajímá, zda existuje větev, po které pokud pojedeme, tak vyhrajeme.
- Tato větev se pozná tak, že ve všech synech jejího koncového vrcholu existuje vyhrávající cesta, tedy ...
- buďto vyhrajeme v prvním synu, nebo ve druhém synu, nebo ve třetím...

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.
- V tomto stromě nás zajímá, zda existuje větev, po které pokud pojedeme, tak vyhrajeme.
- Tato větev se pozná tak, že ve všech synech jejího koncového vrcholu existuje vyhrávající cesta, tedy ...
- buďto vyhrajeme v prvním synu, nebo ve druhém synu, nebo ve třetím...
- V k-tém synu vyhrajeme, jestliže protihráč prohraje v prvním synu a současně ve druhém synu a současně ve třetím synu... tohoto stavu.

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.
- V tomto stromě nás zajímá, zda existuje větev, po které pokud pojedeme, tak vyhrajeme.
- Tato větev se pozná tak, že ve všech synech jejího koncového vrcholu existuje vyhrávající cesta, tedy ...
- buďto vyhrajeme v prvním synu, nebo ve druhém synu, nebo ve třetím...
- V k-tém synu vyhrajeme, jestliže protihráč prohraje v prvním synu a současně ve druhém synu a současně ve třetím synu... tohoto stavu.
- Prohraje tam, jestliže my (pro dotyčný stav) umíme vyhrát buďto v prvním synu, nebo ve druhém, anebo ve třetím...

AND-OR stromy

- Máme-li graf konečné hry, můžeme z něj postavit strom odpovídající hře.
- V tomto stromě nás zajímá, zda existuje větev, po které pokud pojedeme, tak vyhrajeme.
- Tato větev se pozná tak, že ve všech synech jejího koncového vrcholu existuje vyhrávající cesta, tedy ...
- buďto vyhrajeme v prvním synu, nebo ve druhém synu, nebo ve třetím...
- V k-tém synu vyhrajeme, jestliže protihráč prohraje v prvním synu a současně ve druhém synu a současně ve třetím synu... tohoto stavu.
- Prohraje tam, jestliže my (pro dotyčný stav) umíme vyhrát buďto v prvním synu, nebo ve druhém, anebo ve třetím...
- Podmínky AND a OR se stále střídají, proto AND-OR strom.

Hry s ohodnocením

Definition

Hra s ohodnocením je taková hra, kdy cílové stavy jsou ohodnoceny číslem. Jeden hráč se pokouší výsledek maximalizovat, druhý minimalizovat.

Definition

Hra s nulovým součtem je taková hra, ve které zisk jednoho hráče je roven ztrátě druhého hráče.

Některé hry

- **Výlet s přítelkyní do New Yorku:** Chceme navštívit co nejvíce hostinců a technických pamětihodností, přítelkyně chce vidět co nejvíce muzeí a kadeřnictví. Dohodnete se tudíž, že se budete střídat v rozhodování kam jít na jednotlivých křižovatkách.

Některé hry

- **Výlet s přítelkyní do New Yorku:** Chceme navštívit co nejvíce hostinců a technických pamětihodností, přítelkyně chce vidět co nejvíce muzeí a kadeřnictví. Dohodnete se tudíž, že se budete střídat v rozhodování kam jít na jednotlivých křižovatkách.
- **Traverzování po matici:** První hráč mění sloupce, druhý hráč mění sloupce. Začínáme v prvním řádku, první hráč vybere sloupec v prvním řádku a hodnotu na jeho pozici získává. Druhý hráč vybere řádek a získává hodnotu z vybraného řádku ve sloupci vybraném prvním hráčem. Takto se střídají (předem známou dobu).

Některé hry

- **Výlet s přítelkyní do New Yorku:** Chceme navštívit co nejvíce hostinců a technických pamětihodností, přítelkyně chce vidět co nejvíce muzeí a kadeřnictví. Dohodnete se tudíž, že se budete střídat v rozhodování kam jít na jednotlivých křižovatkách.
- **Traverzování po matici:** První hráč mění sloupce, druhý hráč mění sloupce. Začínáme v prvním řádku, první hráč vybere sloupec v prvním řádku a hodnotu na jeho pozici získává. Druhý hráč vybere řádek a získává hodnotu z vybraného řádku ve sloupci vybraném prvním hráčem. Takto se střídají (předem známou dobu).
- **Společná otázka:** Jak hrát?

Algoritmus MINIMAX

- Algoritmus lze použít pro hry s ohodnocením.

Algoritmus MINIMAX

- Algoritmus lze použít pro hry s ohodnocením.
- Postavíme strom hry.

Algoritmus MINIMAX

- Algoritmus lze použít pro hry s ohodnocením.
- Postavíme strom hry.
- Začneme od koncových vrcholů.

Algoritmus MINIMAX

- Algoritmus lze použít pro hry s ohodnocením.
- Postavíme strom hry.
- Začneme od koncových vrcholů.
- Hodnota podstromu je minimum resp. maximum z hodnot synů (podle toho, zda hraje minimalizující nebo maximalizující hráč).

Algoritmus NEGAMAX

- Varianta algoritmu MINIMAX pro hry s nulovým součtem:

$$\max_{i \in S} -f(i) = \min_{i \in S} f(i).$$

Algoritmus NEGAMAX

- Varianta algoritmu MINIMAX pro hry s nulovým součtem:

$$\max_{i \in S} -f(i) = \min_{i \in S} f(i).$$

- Jde vlastně o totéž, je ovšem jednodušší na naprogramování.

Heuristiky

- Obvykle se pokoušíme neprohledávat zbytečně všechno, pokud najdeme jednu možnost výhry, nemusíme hledat i všechny ostatní.

Heuristiky

- Obvykle se pokoušíme neprohledávat zbytečně všechno, pokud najdeme jednu možnost výhry, nemusíme hledat i všechny ostatní.
- α-β-prořezávání: Umíme-li v nějakém synu S vyhrát aspoň α a najdeme v některém následujícím synu T, že protihráč nás umí dotlačit na méně, nemá smysl vrchol T dále zkoumat.

Heuristiky

- Obvykle se pokoušíme neprohledávat zbytečně všechno, pokud najdeme jednu možnost výhry, nemusíme hledat i všechny ostatní.
- α-β-prořezávání: Umíme-li v nějakém synu S vyhrát aspoň α a najdeme v některém následujícím synu T, že protihráč nás umí dotlačit na méně, nemá smysl vrchol T dále zkoumat.
- Pro opačný případ se používá β: Pokud nás nepřítel umí zatlačit na nejvýš β a v jiném synu mu utečeme přes, nemá smysl ten druhý syn zkoumat dále.

Reálné hry

Šachy, dáma, halma, mlýn...

- Můžeme postavit strom hry, ten je ale příliš velký.

Reálné hry

Šachy, dáma, halma, mlýn...

- Můžeme postavit strom hry, ten je ale příliš velký.
- Nasadíme proto všelijaké heuristiky. Ty dosavadní ale stejně daleko nevedou.

Reálné hry

Šachy, dáma, halma, mlýn...

- Můžeme postavit strom hry, ten je ale příliš velký.
- Nasadíme proto všelijaké heuristiky. Ty dosavadní ale stejně daleko nevedou.
- Statická ohodnocovací funkce: Funkce, která se pokouší odhadnout, zda je pozice perspektivní (dobrá) nebo ne.

Reálné hry

Šachy, dáma, halma, mlýn...

- Můžeme postavit strom hry, ten je ale příliš velký.
- Nasadíme proto všelijaké heuristiky. Ty dosavadní ale stejně daleko nevedou.
- Statická ohodnocovací funkce: Funkce, která se pokouší odhadnout, zda je pozice perspektivní (dobrá) nebo ne.
- Prohledáváme strom hry jen po nějakou dobu (do nějaké hloubky). Na nalezené (neterminální) pozice nasadíme statickou ohodnocovací funkci.

Reálné hry

Šachy, dáma, halma, mlýn...

- Můžeme postavit strom hry, ten je ale příliš velký.
- Nasadíme proto všelijaké heuristiky. Ty dosavadní ale stejně daleko nevedou.
- Statická ohodnocovací funkce: Funkce, která se pokouší odhadnout, zda je pozice perspektivní (dobrá) nebo ne.
- Prohledáváme strom hry jen po nějakou dobu (do nějaké hloubky). Na nalezené (neterminální) pozice nasadíme statickou ohodnocovací funkci.
- U šachů například můžeme počítat materiální převahu a body za ohrožené figurky (Colossus na Atari kolem roku 1985).

Horizont, statická ohodnocovací funkce

- Horizont stanoví, do jaké hloubky graf hry (zpravidla realizovaný stromem) prohledáváme.

Horizont, statická ohodnocovací funkce

- Horizont stanoví, do jaké hloubky graf hry (zpravidla realizovaný stromem) prohledáváme.
- Statická ohodnocovací funkce nastoupí, pokud se dostaneme na horizont.

Horizont, statická ohodnocovací funkce

- Horizont stanoví, do jaké hloubky graf hry (zpravidla realizovaný stromem) prohledáváme.
- Statická ohodnocovací funkce nastoupí, pokud se dostaneme na horizont.
- Dalšího zrychlení lze (zkusit) dosáhnout tak, že napřed prohledáváme perspektivní vrcholy (kde statická ohodnocovací funkce dává lepší výsledky).

Horizont, statická ohodnocovací funkce

- Horizont stanoví, do jaké hloubky graf hry (zpravidla realizovaný stromem) prohledáváme.
- Statická ohodnocovací funkce nastoupí, pokud se dostaneme na horizont.
- Dalšího zrychlení lze (zkusit) dosáhnout tak, že napřed prohledáváme perspektivní vrcholy (kde statická ohodnocovací funkce dává lepší výsledky).
- Jde ovšem jen o heuristiku, která někdy funguje, jindy se dostane do problémů!

Komentář k reálným hrám

- Typicky stavíme strom hry. Graf stavíme až v závěrečné fázi hry, do té doby budujeme strom (a ignorujeme možnost, že některé stavy se už vyskytly).

Komentář k reálným hrám

- Typicky stavíme strom hry. Graf stavíme až v závěrečné fázi hry, do té doby budujeme strom (a ignorujeme možnost, že některé stavy se už vyskytly).
- Heuristické algoritmy lze pojmout dvěma způsoby:

Komentář k reálným hrám

- Typicky stavíme strom hry. Graf stavíme až v závěrečné fázi hry, do té doby budujeme strom (a ignorujeme možnost, že některé stavy se už vyskytly).
- Heuristické algoritmy lze pojmout dvěma způsoby:
- Metoda, která se pokouší najít optimum co nejrychleji,

Komentář k reálným hrám

- Typicky stavíme strom hry. Graf stavíme až v závěrečné fázi hry, do té doby budujeme strom (a ignorujeme možnost, že některé stavy se už vyskytly).
- Heuristické algoritmy lze pojmout dvěma způsoby:
- Metoda, která se pokouší najít optimum co nejrychleji,
- metoda, jak najít aspoň nějaké (suboptimální) řešení.

Komentář k reálným hrám

- Typicky stavíme strom hry. Graf stavíme až v závěrečné fázi hry, do té doby budujeme strom (a ignorujeme možnost, že některé stavy se už vyskytly).
- Heuristické algoritmy lze pojmout dvěma způsoby:
- Metoda, která se pokouší najít optimum co nejrychleji,
- metoda, jak najít aspoň nějaké (suboptimální) řešení.
- Zatím bylo to první. Jak použít heuristiku k nalezení suboptimálního řešení?

$\alpha - \beta$ prořezávání jako heuristika

- Jsou dva možné způsoby:

$\alpha - \beta$ prořezávání jako heuristika

- Jsou dva možné způsoby:
- Metoda okénka: Stanovíme krajní hodnoty α a β a výsledky ležící mimo tento interval ořežeme.

$\alpha - \beta$ prořezávání jako heuristika

- Jsou dva možné způsoby:
- Metoda okénka: Stanovíme krajní hodnoty α a β a výsledky ležící mimo tento interval ořežeme.
- Kaskádní varianta – strom rozšiřujeme po hladinách (protože pokud bychom ho stavěli prohledáváním do hloubky, prozkoumáme typicky nezajímavé větve a zajímavé tahy nám uniknou).

Minimaxové věty

- Vraťme se k maticovým hrám (stylu Al-Capone a Babinský na sebe udávají).

Minimaxové věty

- Vraťme se k maticovým hrám (stylu Al-Capone a Babinský na sebe udávají).
- Má smysl uvažovat nejen o deterministické variantě (tzv. čistá strategie – obzvlášť pokud hráči táhnou nezávisle, tedy na sebe nevidí), ale má smysl definovat nějakou pravděpodobnostní distribuci (a podle té hrát). Tomu říkáme mixovaná strategie.

Minimaxové věty

- Vraťme se k maticovým hrám (stylu Al-Capone a Babinský na sebe udávají).
- Má smysl uvažovat nejen o deterministické variantě (tzv. čistá strategie – obzvlášť pokud hráči táhnou nezávisle, tedy na sebe nevidí), ale má smysl definovat nějakou pravděpodobnostní distribuci (a podle té hrát). Tomu říkáme mixovaná strategie.

Minimaxové věty

- Vraťme se k maticovým hrám (stylu Al-Capone a Babinský na sebe udávají).
- Má smysl uvažovat nejen o deterministické variantě (tzv. čistá strategie – obzvlášť pokud hráči táhnou nezávisle, tedy na sebe nevidí), ale má smysl definovat nějakou pravděpodobnostní distribuci (a podle té hrát). Tomu říkáme mixovaná strategie.

Theorem

Pro každou kombinatorickou hru s nulovým součtem s konečnými strategiemi existuje hodnota V a mixovaná strategie pro každého hráče taková, že:

Minimaxové věty

- Vraťme se k maticovým hrám (stylu Al-Capone a Babinský na sebe udávají).
- Má smysl uvažovat nejen o deterministické variantě (tzv. čistá strategie – obzvlášť pokud hráči táhnou nezávisle, tedy na sebe nevidí), ale má smysl definovat nějakou pravděpodobnostní distribuci (a podle té hrát). Tomu říkáme mixovaná strategie.

Theorem

Pro každou kombinatorickou hru s nulovým součtem s konečnými strategiemi existuje hodnota V a mixovaná strategie pro každého hráče taková, že:

- *Pokud podle své strategie hraje druhý hráč, první hráč nemůže vyhrát více než V.*

Minimaxové věty

- Vraťme se k maticovým hrám (stylu Al-Capone a Babinský na sebe udávají).
- Má smysl uvažovat nejen o deterministické variantě (tzv. čistá strategie – obzvlášť pokud hráči táhnou nezávisle, tedy na sebe nevidí), ale má smysl definovat nějakou pravděpodobnostní distribuci (a podle té hrát). Tomu říkáme mixovaná strategie.

Theorem

Pro každou kombinatorickou hru s nulovým součtem s konečnými strategiemi existuje hodnota V a mixovaná strategie pro každého hráče taková, že:

- *Pokud podle své strategie hraje druhý hráč, první hráč nemůže vyhrát více než V.*
- *Pokud podle své strategie hraje první hráč, druhý hráč nemůže vyhrát více než $-V$.*

Nash equilibrium

Definition

Nashovou rovhováhou nazveme sadu mixovaných strategií (pro každého hráče jednu) v konečných hrách aspoň dvou nespolupracujících hráčů, kde žádný z hráčů si nemůže pomoci tím, že strategii změní.

Theorem (J. Nash)

Každá hra n hráčů, kde každý hráč má konečně možných strategií existují strategie určující Nashovu rovhováhu.

Quicksort – ponaučení – metoda rozděl a panuj

- Rozdělenému nepříteli se lépe vládne.

Quicksort – ponaučení – metoda rozděl a panuj

- Rozdělenému nepříteli se lépe vládne.
- Rozděl a panuj rozdělí instanci na přesně definované menší instance,

Quicksort – ponaučení – metoda rozděl a panuj

- Rozdělenému nepříteli se lépe vládne.
- Rozděl a panuj rozdělí instanci na přesně definované menší instance,
- vyřeší každou zvlášť a z jejich řešení zjistí řešení původní instance.

Quicksort – ponaučení – metoda rozděl a panuj

- Rozdělenému nepříteli se lépe vládne.
- Rozděl a panuj rozdělí instanci na přesně definované menší instance,
- vyřeší každou zvlášť a z jejich řešení zjistí řešení původní instance.
- Příklady: Řízení podniku (ředitel leteckých závodů zpravidla neřeší, jak přinýtovat zadní část křídla),

Quicksort – ponaučení – metoda rozděl a panuj

- Rozdělenému nepříteli se lépe vládne.
- Rozděl a panuj rozdělí instanci na přesně definované menší instance,
- vyřeší každou zvlášť a z jejich řešení zjistí řešení původní instance.
- Příklady: Řízení podniku (ředitel leteckých závodů zpravidla neřeší, jak přinýtovat zadní část křídla),
- Příklady (algoritmické): Quicksort, binární vyhledávání.

Metody analýzy složitosti problémů typu rozděl a panuj

- Složitost zpravidla získáme ve tvaru rekurentní formule $T(n) = T(n_1) + T(n_2) + ... + T(n_k) + f(n)$.

Metody analýzy složitosti problémů typu rozděl a panuj

- Složitost zpravidla získáme ve tvaru rekurentní formule $T(n) = T(n_1) + T(n_2) + ... + T(n_k) + f(n)$.
- My chceme rekurenci rozbít. Jedna z metod je indukce.

Metody analýzy složitosti problémů typu rozděl a panuj

- Složitost zpravidla získáme ve tvaru rekurentní formule $T(n) = T(n_1) + T(n_2) + ... + T(n_k) + f(n)$.
- My chceme rekurenci rozbít. Jedna z metod je indukce.
- Příklad: $T(n) = T(\frac{n}{2}) + k$ (binární vyhledání)

Metody analýzy složitosti problémů typu rozděl a panuj

- Složitost zpravidla získáme ve tvaru rekurentní formule $T(n) = T(n_1) + T(n_2) + ... + T(n_k) + f(n)$.
- My chceme rekurenci rozbít. Jedna z metod je indukce.
- Příklad: $T(n) = T(\frac{n}{2}) + k$ (binární vyhledání)
- Co když v Quicksortu jako pivot vybereme medián?

Metody analýzy složitosti problémů typu rozděl a panuj

- Složitost zpravidla získáme ve tvaru rekurentní formule $T(n) = T(n_1) + T(n_2) + ... + T(n_k) + f(n)$.
- My chceme rekurenci rozbít. Jedna z metod je indukce.
- Příklad: $T(n) = T(\frac{n}{2}) + k$ (binární vyhledání)
- Co když v Quicksortu jako pivot vybereme medián?
- V tom případě máme rekurenci: $T(n) = 2T(\frac{n}{2}) + kn$.

Metody analýzy složitosti problémů typu rozděl a panuj

- Složitost zpravidla získáme ve tvaru rekurentní formule $T(n) = T(n_1) + T(n_2) + ... + T(n_k) + f(n)$.
- My chceme rekurenci rozbít. Jedna z metod je indukce.
- Příklad: $T(n) = T(\frac{n}{2}) + k$ (binární vyhledání)
- Co když v Quicksortu jako pivot vybereme medián?
- V tom případě máme rekurenci: $T(n) = 2T(\frac{n}{2}) + kn$.
- Nelze použít jinou metodu?

Master theorem (symetrická verze)

Theorem

Je-li $T(1) = c$ *a* $T(n) = a \cdot T(\frac{n}{b}) + \Theta(n^d)$, *kde* $a \geq 1$, $b > 1$, $d \geq 0$ *a* a, b *přirozená čísla, potom platí:*

- $T(n) \in \Theta(n^d)$, *pokud* $a < b^d$,
- $T(n) \in \Theta(n^d \log n)$, *pokud* $a = b^d$
- *a* $T(n) \in \Theta(n^{\log_b a})$, *pokud* $a > b^d$.

Master theorem – myšlenka důkazu:

- Výpočet si představíme "po hladinách" podle hloubky rekurze.
- Zjistíme složitost každé hladiny zvlášť,
- zjistíme, která bude trvat nejdéle,
- posčítáme.
- Hladin bude $\log_b n$, protože čekáme, až z n zbyde v argumentu jen konstanta.
- Jaká je složitost hladiny k?: $a^k \cdot (\frac{n}{b^k})^d$.

- Která hladina bude trvat nejdéle?
- První (pokud $\left(\frac{a}{b^d}\right)^k < 1$), což je první případ.
- Nebo poslední (pokud $\left(\frac{a}{b^d}\right)^k > 1$), což je třetí případ.
- Nebo všechny stejně (pokud $\left(\frac{a}{b^d}\right)^k = 1$), což je druhý případ.
- Složitost je tedy součtem geometrické posloupnosti!
- Kvocient této řady je menší než jedna, větší než jedna, nebo právě jedna.
- Je-li kvocient různý od jedné, odhadneme součet největším z členů, zbytek je konstanta.
- Je-li kvocient jedna, součet je: hloubka krát složitost libovolné hladiny. A hloubka je $\log n$.

Analýza master-theorem:

- Binární vyhledání: $T(n) = T(\frac{n}{2}) + \Theta(n^0)$
- druhý případ $a = 1, b = 2, d = 0, a = b^d$, tedy složitost binárního vyhledání je $\Theta(\log n)$.
- Quicksort obecně: $T(n) = T(n_1) + T(n - n_1) + \Theta(n^1)$. Master theorem mlčí!
- Quicksort vybereme-li za pivot medián a umíme-li medián vybrat s lineární složitostí: $T(n) = 2T(\frac{n}{2}) + \Theta(n^1)$. Jde opět o druhý případ $a = 2, b = 2, d = 1, a = b^d$, získáváme opět složitost $\Theta(n \log n)$.

Další příklad – násobení matic:

- $A = \begin{pmatrix} A_{11}, A_{12} \\ A_{21}, A_{22} \end{pmatrix}$, $B = \begin{pmatrix} B_{11}, B_{12} \\ B_{21}, B_{22} \end{pmatrix}$, $C = \begin{pmatrix} C_{11}, C_{12} \\ C_{21}, C_{22} \end{pmatrix}$.

Další příklad – násobení matic:

- $A = \begin{pmatrix} A_{11}, A_{12} \\ A_{21}, A_{22} \end{pmatrix}$, $B = \begin{pmatrix} B_{11}, B_{12} \\ B_{21}, B_{22} \end{pmatrix}$, $C = \begin{pmatrix} C_{11}, C_{12} \\ C_{21}, C_{22} \end{pmatrix}$.
- cíl:

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$
$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$
$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$
$$C_{22} = A_{21}B_{12} + A_{22}B_{22}.$$

Další příklad – násobení matic:

- $A = \begin{pmatrix} A_{11}, A_{12} \\ A_{21}, A_{22} \end{pmatrix}$, $B = \begin{pmatrix} B_{11}, B_{12} \\ B_{21}, B_{22} \end{pmatrix}$, $C = \begin{pmatrix} C_{11}, C_{12} \\ C_{21}, C_{22} \end{pmatrix}$.
- cíl:
  $C_{11} = A_{11}B_{11} + A_{12}B_{21}$
  $C_{12} = A_{11}B_{12} + A_{12}B_{22}$
  $C_{21} = A_{21}B_{11} + A_{22}B_{21}$
  $C_{22} = A_{21}B_{12} + A_{22}B_{22}$.
- Trik: Definujeme matice $M_1, \ldots M_7$, aby $C_{11} \ldots C_{22}$ šlo vyjádřit jako jejich součet resp. rozdíl.

Další příklad – násobení matic:

Strassenuv algoritmus

- $M_1 = (A_{11} + A_{22})(B_{11} + B_{22})$,
  $M_2 = (A_{21} + A_{22})B_{11}$,
  $M_3 = A_{11}(B_{12} - B_{22})$,
  $M_4 = A_{22}(B_{21} - B_{11})$,
  $M_5 = (A_{11} + A_{12})B_{22}$,
  $M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$,
  $M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$.

Další příklad – násobení matic:

Strassenuv algoritmus

- $M_1 = (A_{11} + A_{22})(B_{11} + B_{22})$,
  $M_2 = (A_{21} + A_{22})B_{11}$,
  $M_3 = A_{11}(B_{12} - B_{22})$,
  $M_4 = A_{22}(B_{21} - B_{11})$,
  $M_5 = (A_{11} + A_{12})B_{22}$,
  $M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$,
  $M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$.
- $C_{11} = M_1 + M_4 - M_5 + M_7$,
  $C_{12} = M_3 + M_5$,
  $C_{21} = M_2 + M_4$,
  $C_{22} = M_1 - M_2 + M_3 + M_6$.

Další příklad – násobení matic:

- Naivní algoritmus: $T(n) = \Theta(n^3)$.
- Strassenův algoritmus použije jen 7 násobení matic o polovinu menších,
- režie každé iterace je kvadratická (vůči šířce matice),
- rekurence tedy vychází: $T(n) = 7T(\frac{n}{2}) + \Theta(n^2)$.
- Jde o 3. případ ($a = 7, b = 2, d = 2, a > b^d$) a tedy složitost Strassenova algoritmu je: $\Theta(n^{\log_2 7})$.
- Dokazujte toto indukcí...

Konec

...děkuji za pozornost...

Otázky?