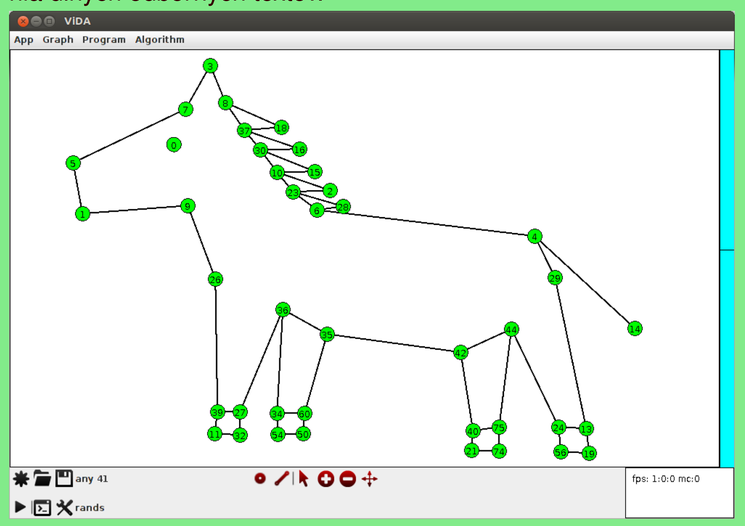
ViDA: Vizualizácia distribuovaných algoritmov

Michal Anderle[1], **Ján Hozza**[1]
Supervisor: Jakub Kováč[1]
[1] Katedra informatiky FMFI UK, Mlynská Dolina, 842 48 Bratislava

Úvod

V našom ročníkovom projekte, sme sa rozhodli zaoberať vizualizáciou distribuovaných algortimov pomocou Java aplikácie, ktorá umožňuje jednoduché a rýchle pochopenie tématu, bez študovania dlhých odborných textov.



Hlavné ciele

- vizualizácie ušité na mieru konkrétnym distribuovaným algoritmov
- interaktivita s používateľom
- prehľadnosť a jednoduchosť používania aplikácie
- schopnosť vizualizovať vlastné algoritmy

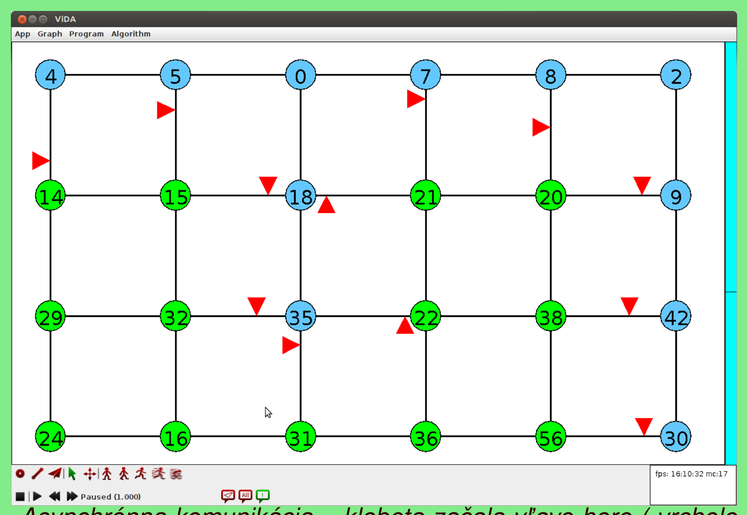
Distribuované algoritmy

Vlastnosti modelu:

- niekoľko počítačov zapojených do siete obojsmernými linkami
- majú jednoznačné id, komunikujú len správami
- správy sa nestrácajú, nemenia poradie, ale môže im to trvať ľubovoľne dlho – asynchrónna komunikácia

Ciele:

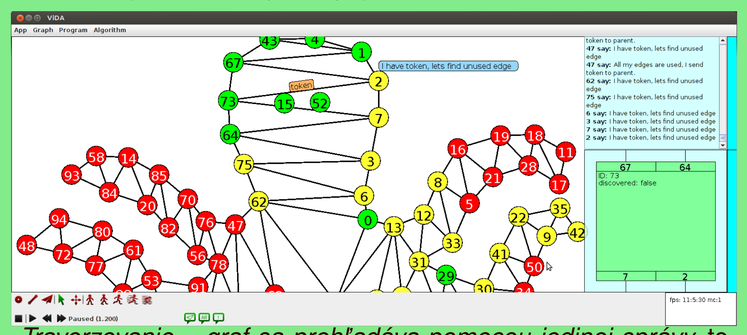
- poslať čo najmenej správ
- minimalizovať dobu behu algoritmu



Asynchrónna komunikácia – klebeta začala vľavo hore (vrchole 4) a dostala sa na druhý koniec mriežky (vrchol 30) skôr, než do spodného suseda (vrchol 14). Aj toto naša vizualizácia dokáže spraviť.

Navizualizované algoritmy

- broadcast – *ako povedať novú klebetu všetkým v sieti?*
- traverzovanie – *ako len s pomocou jednej správy prehľadať celý graf?*
- voľba šéfa na úplnom grafe – *ako sa spomedzi niekoľkých identických programov dá zvoliť jeden šéf? a čo ak pri tom chceme poslať čo najmenej správ?*



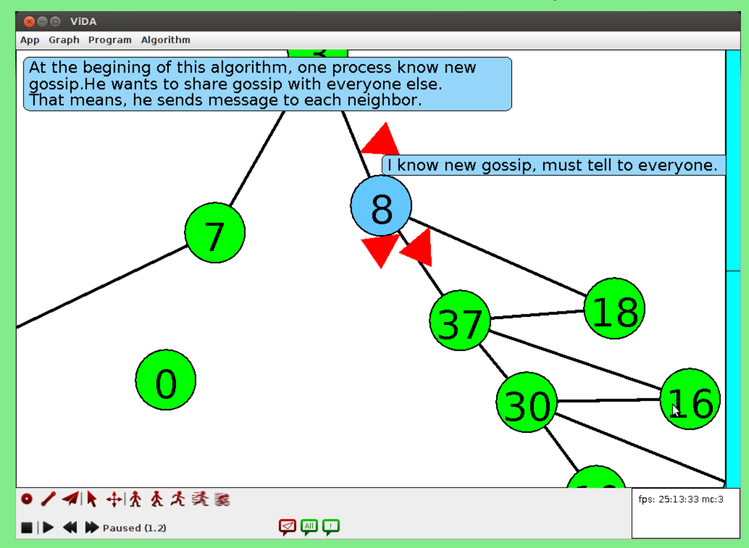
Traverzovanie – graf sa prehľadáva pomocou jedinej správy, tokenu. *Zelené vrcholy sú nenavštívené, oranžové sú navštívené, červené sú úplne vybavené – už preskúmali všetkých susedov*

Nástroje

- úprava grafu
  - pridávanie, mazanie, editovanie, hýbanie vrcholov a hrán
  - ukladanie, automatické generovanie rôznych typov a veľkostí
  - verifikácia – napr. niektoré algoritmy sú určené len pre úplné grafy
- časovanie správ
  - možnosť chytiť správu a presunúť ju na iné miesto na hrane
  - nastavovanie rýchlostí hrán, správ
- selekcia – zobrazovanie dodatočných informácií
- plánujú sa mnohé ďalšie

Spôsob vizualizácie

- informácie priamo v grafe – intuitívne spojenie vizuálnych a hodnotových vlastností, napr. veľkosť vrchola = level, farba vrchola = stav (napr. červený = mŕtvy/porazený/neaktívny)
- zobrazovanie udalostí priamo v grafe – netreba vrtieť hlavou a hľadať, čo sa kde deje, informácie sa zobrazujú tam, kde sa ich to týka, všetko pomocou *vyskakovacích bubliniek*
- interaktivita – *čo by sa stalo, keď. . . ?* užívateľ môže priamo ovplyvňovať, čo sa stane
- detekcia a vystevlenie zaujímavých udalostí – keď sa niečo stane, aplikácia sa pozastaví a vysvetlí *čo sa stalo? prečo sa to stalo? kde sa to stalo? čo sa bude diať ďalej?*



Zobrazovanie informácií v bublinkách. Algoritmus broadcast.

Vlastné vizualizácie

- možnosť vytvárania vlastných vizualizácií
- knižnica vidalib
  - jednoduchá knižnica v C++ komunikujúca s naším programom
  - poskytnutie celej palety nástrojov – vyskakovacie bublinky, zmena farby a veľkosti vrcholov . . .
- hlbšie pochopenie algoritmu, po jeho naimplementovaní
- potešenie z vlastných fungujúcich algoritmov

Plány do budúcnosti

- ďalšie algoritmy
  - GHS – *voľba šéfa na všeobecnom grafe*
  - KKM – *voľba šéfa s využitím traverzovania*

  - routing – *smerovanie dát v sieti. Kam poslať paket, aby sa dostal do cieľa?*
  - problém dohody
- viac zábavy, viac interaktivity – *užívateľ sa môže zahrať na zákeráka a snažiť sa donútiť algoritmus, aby poslal čo najviac správ*
- viacero programovacích jazykov, viac nástrojov pre vizualizáciu