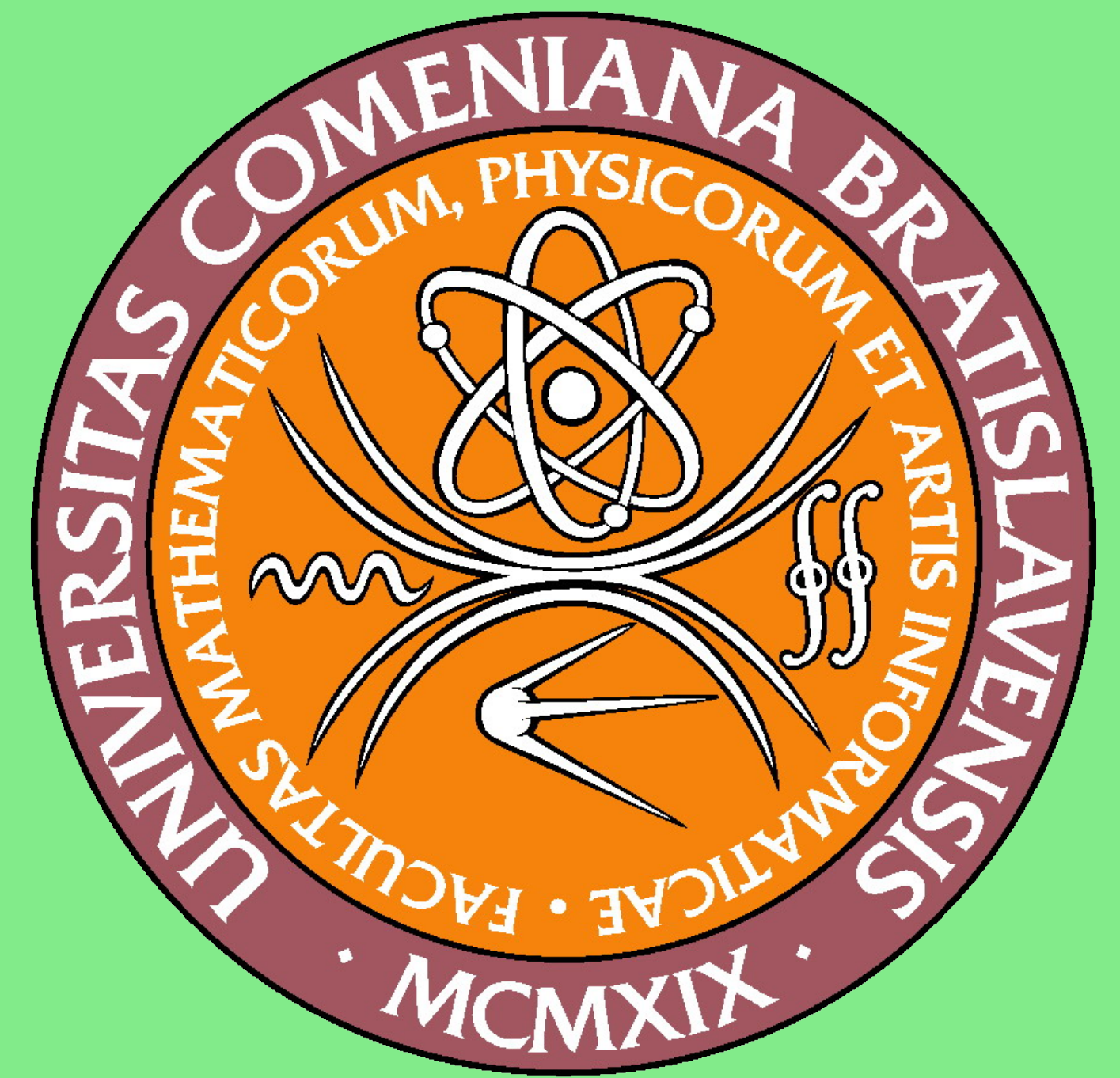


# ViDA: Vizualizácia distribuovaných algoritmov

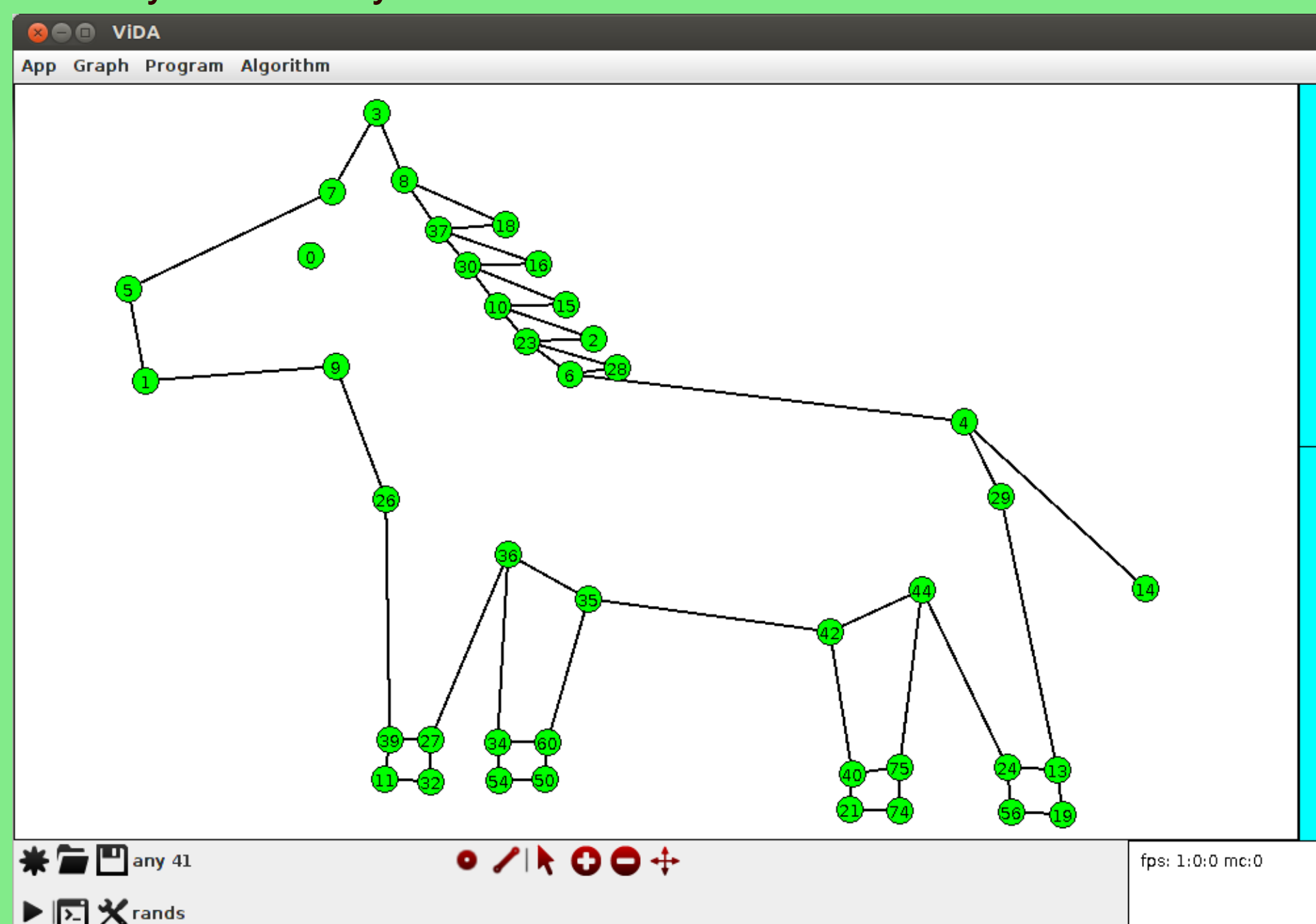
Michal Anderle<sup>1</sup>, Ján Hozza<sup>1</sup>  
Supervisor: Jakub Kováč<sup>1</sup>

<sup>1</sup> 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

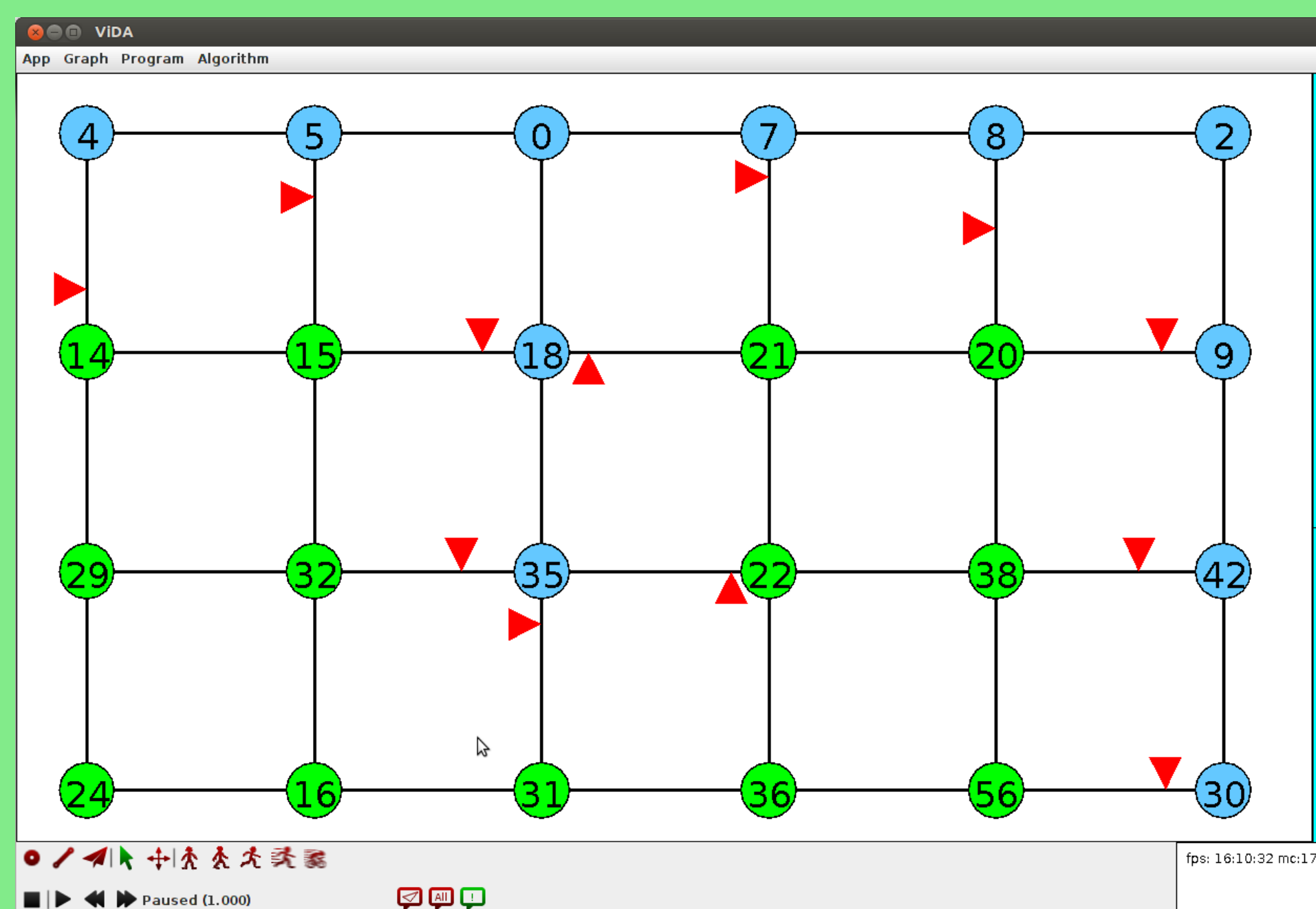
## DISTRIBUOVANÉ ALGORITMY

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:

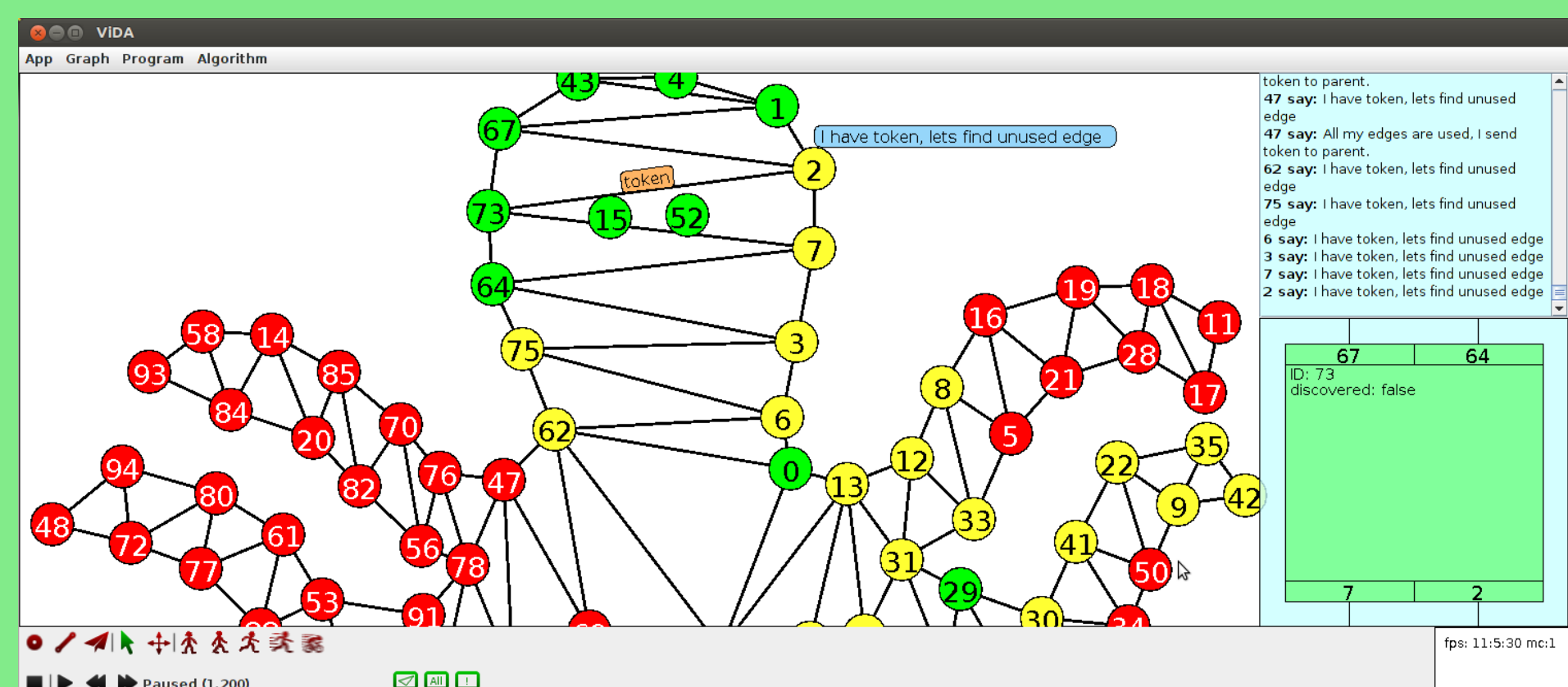
- poslať čo najmenej správ
- minimalizovať dobu behu algoritmu



*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 nenavštivené, oranžové sú navštivené, červené sú úplne vybavené – už preskúmali všetkých susedov*

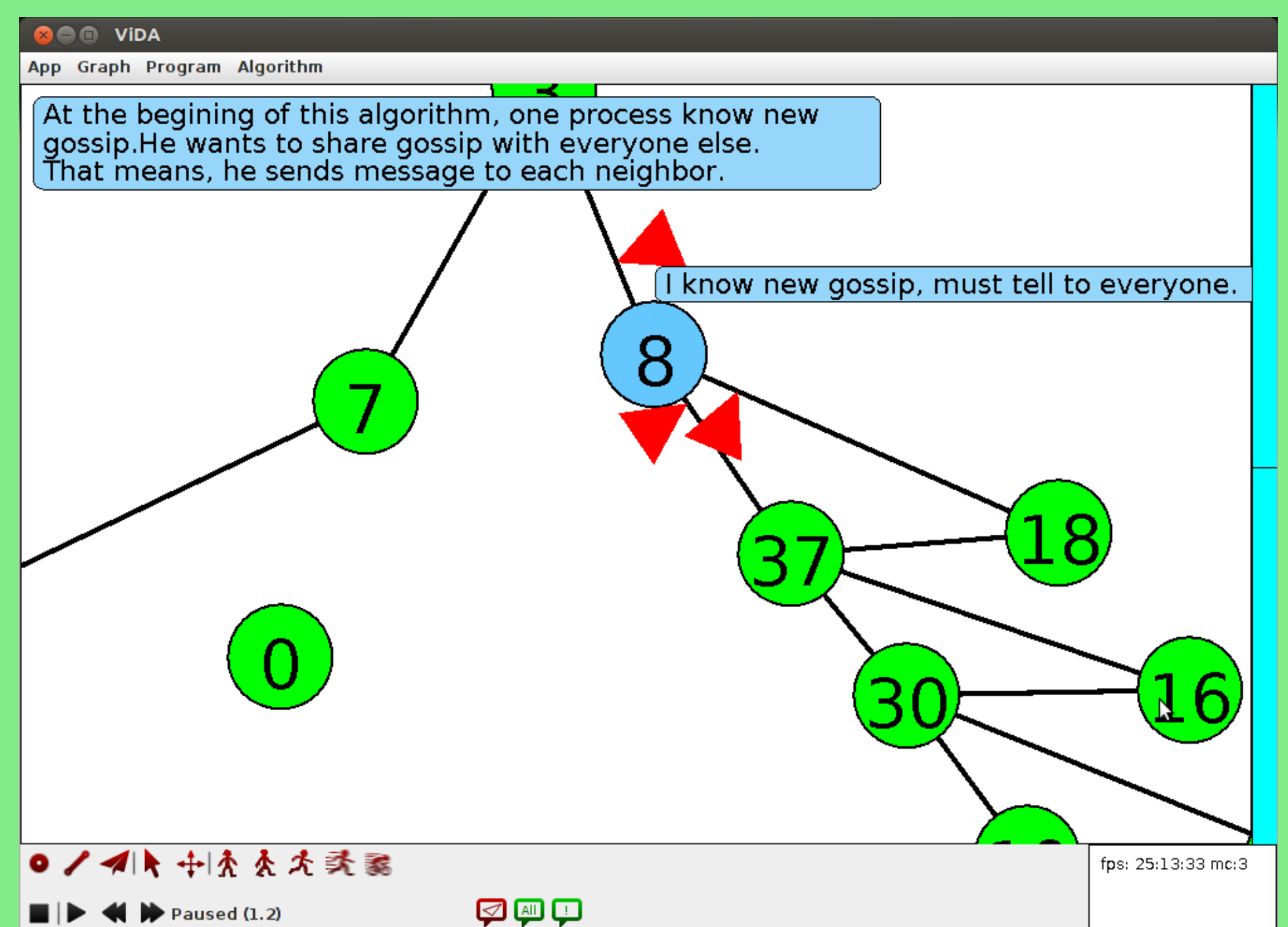
## NÁSTROJE

- 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 bublíniek*
- interaktivita – čo by sa stalo, keď...? užívateľ môže priamo ovplyvňovať, čo sa stane
- detekcia a vystavenie 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 našim 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