

PageRank

aneb lineární algebra za miliony

Tomáš Svoboda, svobodat@fel.cvut.cz
první verze přednášky Ondřej Drbohlav, drbohlav@fel.cvut.cz
<http://cw.felk.cvut.cz/doku.php/courses/b4b33rph/start>

10. prosince 2024



<https://www.menti.com/al7b8cnm4ya7>

01

Navigation icons: back, forward, search, and other controls. 2/43

Notes

Vyzkoušejte si své hledání. Co používáte jako hlavní vyhledávač, resp. navigátor na internetu? Napište na slido.

Typické PC

- ▶ 14-ti palcový monitor
- ▶ Procesor: Pentium, 1 jádro, 233MHz
- ▶ 32MB RAM
- ▶ 4.3GB harddisk
- ▶ floppy disk



TS home PC: Intel 386SX, 33MHz (turbo!), 2MB RAM, 105 MB hard, černobílý monitor 14".

Notes

Už jsme si odvykli počítat v MB, není-li pravda?



Notes

Program OI je tu s námi již nějaký čas, takto to bylo na počátku. Ale dnes již nikdo z vás nepamatuje.

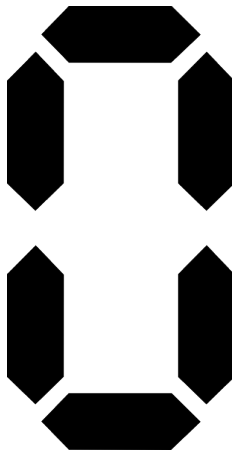
1997 – budoucí studenti OI, jak šel RPH čas



OI

Notes

Program OI je tu s námi již nějaký čas, takto to bylo na počátku. Ale dnes již nikdo z vás nepamatuje.



Notes

Program OI je tu s námi již nějaký čas, takto to bylo na počátku. Ale dnes již nikdo z vás nepamatuje.

1997 – Internet (skoro tak, jak ho známe)

► Věk:



- ▶ první WWW prohlížeč Mosaic (1993)
- ▶ Netscape (1994)

Vyhledávací stroje:

- ▶ Altavista
- ▶ viz pár příkladu ze stroje času <http://web.archive.org/>

Dávná historie internetu

- ▶ <http://web.archive.org/web/19961230033620/http://www.uzdroje.cz/>
- ▶ <http://web.archive.org/web/19961114213839/http://www.seznam.cz/>
- ▶ <http://web.archive.org/web/19970502010243/http://www.feld.cvut.cz/>
- ▶ <http://web.archive.org/web/19970501133354/http://cmp.felk.cvut.cz/>
- ▶ <http://web.archive.org/web/19981111184551/http://google.com/>
- ▶ <http://web.archive.org/web/19981111183552/http://google.stanford.edu/>
- ▶ <http://web.archive.org/web/19990125084553/http://alpha.google.com/>
- ▶ <http://web.archive.org/web/19961020022754/http://www9.yahoo.com/>
- ▶ <http://cmp.felk.cvut.cz/~svoboda/Lodenice/>

Příklad: Altavista



☐ Click Here

AltaVista®

The most powerful and useful guide to the Net

Ask AltaVista™ a question. Or enter a few words in any language [Help](#) - [Advanced](#)

Example: Where can I download mp3 files for instrumental music?

Specialty Searches

[AV Family Filter](#) - [AV Photo Finder](#) - [AV Tools & Gadgets](#)
[Entertainment](#) - [Health](#) - [Online Shopping](#) - [Careers](#) - [Maps](#)
[People Finder](#) - [Stock Quotes](#) - [Travel](#) - [Usenet](#) - [Yellow Pages](#)

CATEGORIES

[Automotive](#)
[Business & Finance](#)
[Computers & Internet](#)
[Health & Fitness](#)
[Hobbies & Interests](#)
[Home & Family](#)
[Media & Amusements](#)
[People & Chat](#)
[Reference & Education](#)
[Shopping & Services](#)
[Society & Politics](#)
[Sports & Recreation](#)
[Travel & Vacations](#)

NEWS BY ABCNEWS.com

- ▶ [Lewinsky Talks](#)
- ▶ [Olympic House-cleaning](#)
- ▶ [Jasper Trial Begins](#)
- ▶ [Papal Mass Draws 1 Million Mexicans](#)

ALTAVISTA HIGHLIGHTS

Search Clinton Video Footage:



- ▶ [New Stat of the Union](#)
- ▶ [Impeach](#)
- ▶ [Clinton Testimony](#)

Video courtesy of C-SPAN.

Notes

OTHER SERVICES

[AltaVista Discovery](#) - [Video Search Demo](#)
[FREE Email](#) - [AV Translation Services](#)
[Make Us Your Homepage](#) - [Create A Card](#)
[Photo Albums!](#) - [Asian Languages](#)

Featured Sponsors

[50% Savings!](#)
[Quality DutyFree Jewelry!](#)
[Great Gifts from BLOCKBUSTER®](#)
♦ [Save on bestsellers everyday at Amazon!](#)

Příklad: Hledání

Hledej: Apple

1. ... byla jsem za Petrou a ta mě požádala o recept na skvělý babiččin koláč. Tady je: jedno **jablko**, dvě vejce, ...

2. ...

⋮

105. **Apple**. Chystáme se na výrobu iPodů. Až vyrostete, kupujte to!

⋮

217. **Jablko** patří mezi ovoce. Příbuzným druhem je hruška ...

⋮

- ▶ velice často *nerelevantní* výsledky
- ▶ nic se nedá najít a nebo to zabere spoustu času
- ▶ v dokumentu se slova sice vyskytují, ale dokument sám je na poměrně obskurních stránkách
- ▶ → bookmarks/záložky

Příčiny? Jak se hledalo?

Notes

Odskok: jak se to dělá s vědeckými, inženýrskými články, ale vlastně i například s knihami či jiným tvůrčím dílem obecně:

- Peer review.
- Citační ohlas článku, resp. kolik lidí s knihu koupí.
- Zkouška časem. Je obsah stále relevantní?

- 10/43

Notes

- Peer review.
- Citační ohlas článku, resp. kolik lidí s knihu koupí.
- Zkouška časem. Je obsah stále relevantní?

Obecně: Jak hodnotit tvůrčí díla (knihy, obrazy, filmy, články)? **Menti**

Notes

- Slovem „neškáluje“ rozumíme, že nová díla, resp. weby přibývají rychleji, než je lze ručně organizovat či hodnotit.
- Ostatně podobný problém nastává i u vědeckých článků. Příliš mnoho autorů na příliš málo kvalitních oponentů.

Obecně: Jak hodnotit tvůrčí díla (knihy, obrazy, filmy, články)?

- ▶ Peer review, recenze, ...
 - ▶ Záleží od koho recenze nebo peer review je?
 - ▶ Kolik recenzí, shodnou se na něčem?
 - ▶ Je review/recenze „blind“?

Notes

- Slovem „neškáluje“ rozumíme, že nová díla, resp. weby přibývají rychleji, než je lze ručně organizovat či hodnotit.
- Ostatně podobný problém nastává i u vědeckých článků. Příliš mnoho autorů na příliš málo kvalitních oponentů.

Obecně: Jak hodnotit tvůrčí díla (knihy, obrazy, filmy, články)?

- ▶ Peer review, recenze, ...
 - ▶ Záleží od koho recenze nebo peer review je?
 - ▶ Kolik recenzí, shodnou se na něčem?
 - ▶ Je review/recenze „blind“?
- ▶ Čitační ohlas článku, resp. kolik lidí s knihu koupí.
 - ▶ Cena, dostupnost, ...
 - ▶ Reklama!
 - ▶ Osobnost autor*ky.

Notes

- Slovem „neškáluje“ rozumíme, že nová díla, resp. weby přibývají rychleji, než je lze ručně organizovat či hodnotit.
- Ostatně podobný problém nastává i u vědeckých článků. Příliš mnoho autorů na příliš málo kvalitních oponentů.

Obecně: Jak hodnotit tvůrčí díla (knihy, obrazy, filmy, články)?

- ▶ Peer review, recenze, ...
 - ▶ Záleží od koho recenze nebo peer review je?
 - ▶ Kolik recenzí, shodnou se na něčem?
 - ▶ Je review/recenze „blind“?
- ▶ Citační ohlas článku, resp. kolik lidí s knihu koupí.
 - ▶ Cena, dostupnost, ...
 - ▶ Reklama!
 - ▶ Osobnost autor*ky.
- ▶ Zkouška časem. Je obsah stále relevantní?
 - ▶ Stále se kupuje? Re-edice, ...
 - ▶ Inspiruje jinou oblast? Film, divadlo, ...
 - ▶ Citační ohlas.

Notes

- Slovem „neškáluje“ rozumíme, že nová díla, resp. weby přibývají rychleji, než je lze ručně organizovat či hodnotit.
- Ostatně podobný problém nastává i u vědeckých článků. Příliš mnoho autorů na příliš málo kvalitních oponentů.

Obecně: Jak hodnotit tvůrčí díla (knihy, obrazy, filmy, články)?

- ▶ Peer review, recenze, ...
 - ▶ Záleží od koho recenze nebo peer review je?
 - ▶ Kolik recenzí, shodnou se na něčem?
 - ▶ Je review/recenze „blind“?
- ▶ Citační ohlas článku, resp. kolik lidí s knihu koupí.
 - ▶ Cena, dostupnost, ...
 - ▶ Reklama!
 - ▶ Osobnost autor*ky.
- ▶ Zkouška časem. Je obsah stále relevantní?
 - ▶ Stále se kupuje? Re-edice, ...
 - ▶ Inspiruje jinou oblast? Film, divadlo, ...
 - ▶ Citační ohlas.

Ruční hodnocení stále nejlepší, ale *neškáluje*!

Notes

- Slovem „neškáluje“ rozumíme, že nová díla, resp. weby přibývají rychleji, než je lze ručně organizovat či hodnotit.
- Ostatně podobný problém nastává i u vědeckých článků. Příliš mnoho autorů na příliš málo kvalitních oponentů.

Notes

Co musí vyhledávací stroj umět?

- ▶ **pracovat plně automaticky**
- ▶ procházet Internet
- ▶ indexovat stránky (uchovávat jejich obsah ve formě vhodné k rychlému vyhledání při zadání klíčového slova)
- ▶ hodnotit důležitost stránek

Co musí vyhledávací stroj umět?

- ▶ pracovat plně automaticky
- ▶ **procházet Internet**
- ▶ indexovat stránky (uchovávat jejich obsah ve formě vhodné k rychlému vyhledání při zadání klíčového slova)
- ▶ hodnotit důležitost stránek

Jak procházet web a nezacyklit se?

Co musí vyhledávací stroj umět?

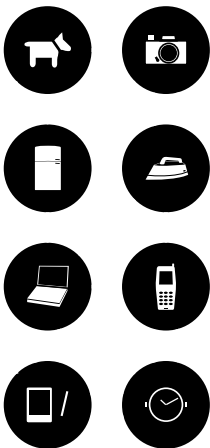
- ▶ pracovat plně automaticky
- ▶ procházet Internet
- ▶ **indexovat stránky (uchovávat jejich obsah ve formě vhodné k rychlému vyhledání při zadání klíčového slova)**
- ▶ hodnotit důležitost stránek

Jaké struktury dat zvolit pro indexování? Jak provádět efektivní vyhledávání dokumentů s výskytem klíčového slova?

Co musí vyhledávací stroj umět?

- ▶ pracovat plně automaticky
 - ▶ procházet Internet
 - ▶ indexovat stránky (uchovávat jejich obsah ve formě vhodné k rychlému vyhledání při zadání klíčového slova)
 - ▶ **hodnotit důležitost stránek**
- ▶ Je přesně to, co vyhledávače v roce 1997 neuměly.
- ▶ Přímo souvisí s pozorovaným problémem s nerelevantností výsledků hledání.
- ▶ Očekávané zlepšení výsledků vyhledávání je velké. Většina textu na internetu se obsahuje cca 10.000 slov; dnešní velikost internetu – miliardy stránek ⇒ seřazení stránek podle důležitosti je opravdu potřebné!

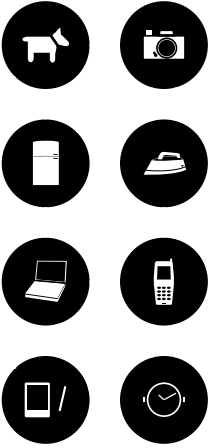
Jak automaticky zjistit důležitost stránek? **Menti**



Notes

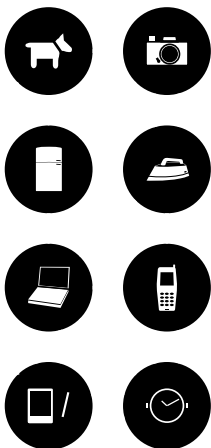
Otázka na Menti.

Jak automaticky zjistit důležitost stránek?



- ▶ velikost stránky?
- ▶ použití spisovného jazyka?
- ▶ kritéria založená na rozpoznávání smyslu textu?
- ▶ ...

Jak automaticky zjistit důležitost stránek?

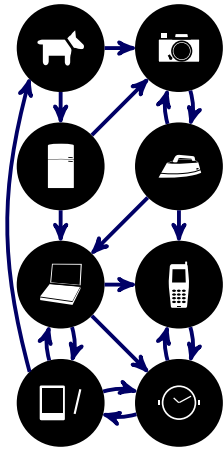


- ▶ velikost stránky?
- ▶ použití spisovného jazyka?
- ▶ kritéria založená na rozpoznávání smyslu textu?
- ▶ ...

Možné problémy:

- ▶ částečně by to mohlo fungovat, ale ...
- ▶ jednoduchá kritéria je “snadné” splnit: *Velikost*: dosáhnout je to podstatně jednodušší než např. ve světě firem [Update2023: pomněte ChatGPT a spol.]
- ▶ *smysl textu*: extrémně těžké, neumí se to
- ▶ leccos je v zásadě snadné zfalšovat

Jak automaticky zjistit důležitost stránek?

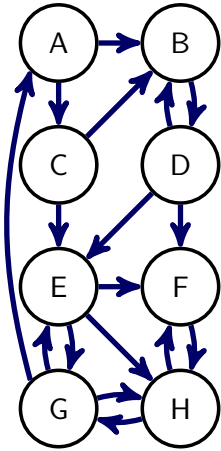


- ▶ web nejsou jen stránky, ale i odkazy mezi nimi
- ▶ mohl bych je použít pro spočítání důležitosti stránek?

Úvahy:

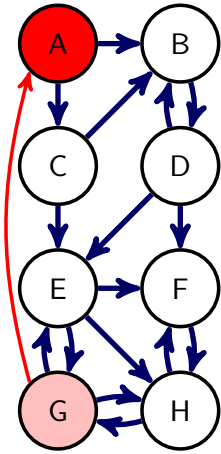
- ▶ pokud **○** doporučuje (odkazuje) mnoho lidí, **○** je věrohodná
- ▶ velká výhoda bude založit skóre stránky na odkazech **na** tuto stránku, tj. **zpětných** odkazech, protože je to něco, co vlastník stránky nemá v moci.

Nápad 1: počítání zpětných odkazů



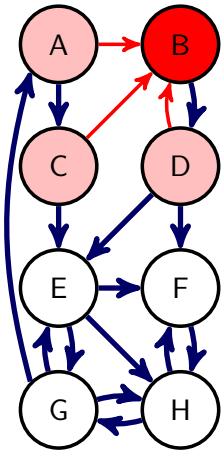
- začneme co nejjednodušeji: co spočítat počet zpětných odkazů?

Nápad 1: počítání zpětných odkazů



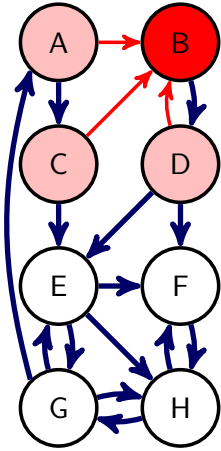
- ▶ začneme co nejjednodušeji: co spočítat počet zpětných linků?
- ▶ $W_A = 1$

Nápad 1: počítání zpětných odkazů



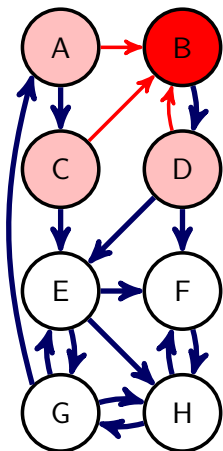
- ▶ začneme co nejjednodušeji: co spočítat počet zpětných linků?
- ▶ $W_A = 1$
- ▶ $W_B = 3$

Nápad 1: počítání zpětných odkazů



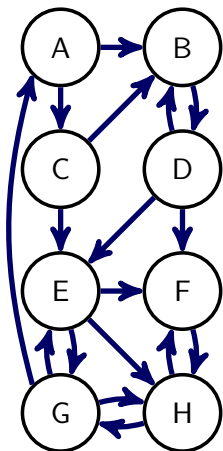
- ▶ začneme co nejjednodušeji: co spočítat počet zpětných linků?
- ▶ $W_A = 1$
- ▶ $W_B = 3$
- ▶ ...
- ▶ $W = [1, 3, 1, 1, 3, 3, 2, 3]$

Nápad 1: počítání zpětných odkazů



- ▶ začněme co nejjednodušší: co spočítat počet zpětných odkazů?
- ▶ $W_A = 1$
- ▶ $W_B = 3$
- ▶ ...
- ▶ $W = [1, 3, 1, 1, 3, 3, 2, 3]$
- ▶ jako první přiblížení je to fajn, ale je samotný počet zpětných odkazů jednak nepopisuje v podstatě vůbec strukturu sítě, jednak je snadno manipulovatelný (návod: pan Zlobivý 🧠 si koupí 10 domén a z každé z nich odkáže svou hlavní doménu)

Nápad 1.1: Váhování zpětných odkazů I

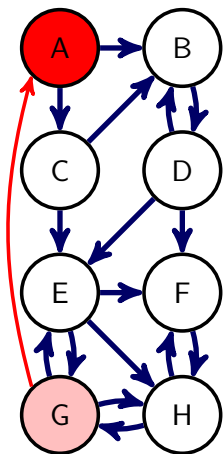


Úvaha:

- ▶ Není jedno, kdo na mě dává odkaz. Odkaz od důležité stránky má větší váhu než víceméně náhodný odkaz.
- ▶ Zkusme tedy sčítat ne *počet* odkazů, ale jejich *váhy*:

Podobná úvaha jako při ručním hodnocení. Osoba recenzenta je důležitá.

Nápad 1.1: Váhování zpětných odkazů I



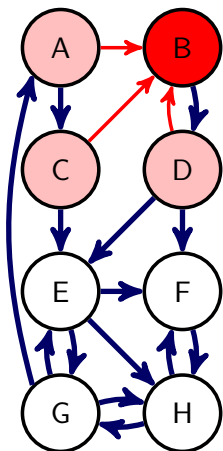
Úvaha:

- ▶ Není jedno, kdo na mě dává odkaz. Odkaz od důležité stránky má větší váhu než víceméně náhodný odkaz.
- ▶ Zkusme tedy sčítat ne *počet* odkazů, ale jejich *váhy*:
- ▶ $W_A = W_G$

Notes

Podobná úvaha jako při ručním hodnocení. Osoba recenzenta je důležitá.

Nápad 1.1: Váhování zpětných odkazů I



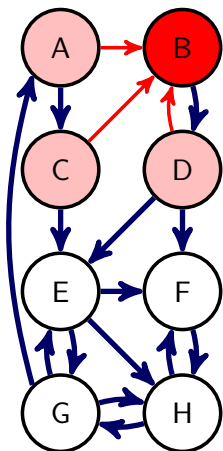
Úvaha:

- ▶ Není jedno, kdo na mě dává odkaz. Odkaz od důležité stránky má větší váhu než víceméně náhodný odkaz.
- ▶ Zkusme tedy sčítat ne *počet* odkazů, ale jejich *váhy*:
- ▶ $W_A = W_G$
- ▶ $W_B = W_A + W_C + W_D$

Notes

Podobná úvaha jako při ručním hodnocení. Osoba recenzenta je důležitá.

Nápad 1.1: Váhování zpětných odkazů I



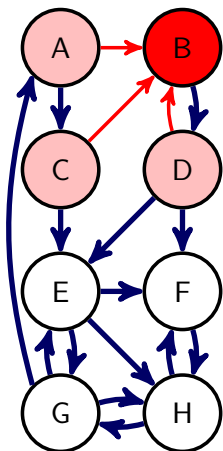
Úvaha:

- ▶ Není jedno, kdo na mě dává odkaz. Odkaz od důležité stránky má větší váhu než víceméně náhodný odkaz.
- ▶ Zkusme tedy sčítat ne *počet* odkazů, ale jejich *váhy*:
- ▶ $W_A = W_G$
- ▶ $W_B = W_A + W_C + W_D$
- ▶ $W_D = W_B$
- ▶ $\vdots$

Notes

Podobná úvaha jako při ručním hodnocení. Osoba recenzenta je důležitá.

Nápad 1.1: Váhování zpětných odkazů I



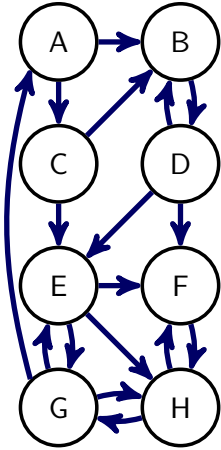
Úvaha:

- ▶ Není jedno, kdo na mě dává odkaz. Odkaz od důležité stránky má větší váhu než víceméně náhodný odkaz.
- ▶ Zkusme tedy sčítat ne *počet* odkazů, ale jejich *váhy*:
- ▶ $W_A = W_G$
- ▶ $W_B = W_A + W_C + W_D$
- ▶ $W_D = W_B$
- ▶ $\vdots$
- ▶ To vypadá rozumně, navíc 🧠 má trochu ztíženou práci: musí myslet na to, že svým „pomocným“ doménám musí zajistit kredibilitu.

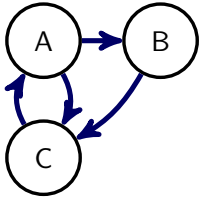
Nápad 1.1: Váhování zpětných odkazů II

Úvaha:

- máme ale problém, uvažme jednoduchou síť



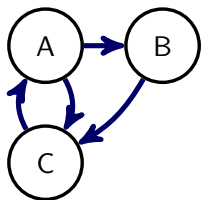
Nápad 1.1: Váhování zpětných odkazů II



Úvaha:

- ▶ máme ale problém, uvažme jednoduchou síť

Nápad 1.1: Váhování zpětných odkazů II



Úvaha:

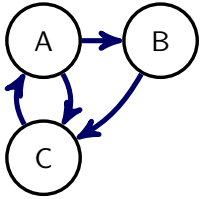
- ▶ máme ale problém, uvažme jednoduchou síť

$$W_A = W_C$$

$$W_B = W_A$$

$$W_C = W_A + W_B$$

Nápad 1.1: Váhování zpětných odkazů II



Úvaha:

- ▶ máme ale problém, uvažme jednoduchou síť

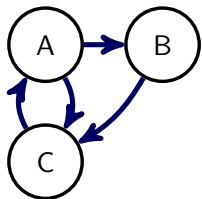
$$W_A = W_C$$

$$W_B = W_A$$

$$W_C = W_A + W_B$$

- ▶ to není možné splnit

Nápad 1.1: Váhování zpětných odkazů II



Úvaha:

- ▶ máme ale problém, uvažme jednoduchou síť

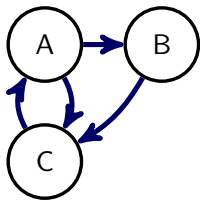
$$W_A = W_C$$

$$W_B = W_A$$

$$W_C = W_A + W_B$$

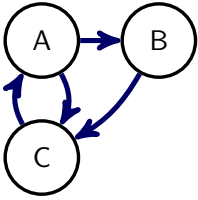
- ▶ to není možné splnit
- ▶ možná není správné, že stránky mohou hlasovat pro libovolný počet ostatních stránek
- ▶ co kdybychom tedy každému dali jen jeden hlas, který může spravedlivě dělit mezi všechny stránky, které doporučuje?

Nápad 1.2: Poměrné hlasování



- ▶ označme n_A počet odkazů vedoucích z webu A
- ▶ $n_A = 2, n_B = 1, n_C = 1$

Nápad 1.2: Poměrné hlasování



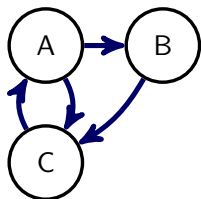
- ▶ označme n_A počet odkazů vedoucích z webu A
- ▶ $n_A = 2, n_B = 1, n_C = 1$
- ▶ s použitím poměrného hlasování tedy bude

$$W_A = \frac{W_C}{n_C} = W_C$$

$$W_B = \frac{W_A}{n_A} = \frac{1}{2} W_A$$

$$W_C = \frac{W_A}{n_A} + \frac{W_B}{n_B} = \frac{1}{2}W_A + W_B$$

Nápad 1.2: Poměrné hlasování



- ▶ označme n_A počet odkazů vedoucích z webu A
- ▶ $n_A = 2$, $n_B = 1$, $n_C = 1$
- ▶ s použitím poměrného hlasování tedy bude

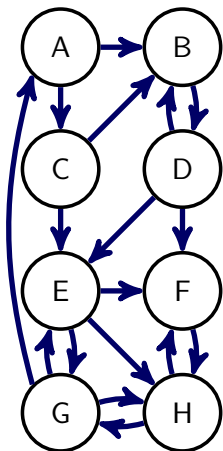
$$W_A = \frac{W_C}{n_C} = W_C$$

$$W_B = \frac{W_A}{n_A} = \frac{1}{2} W_A$$

$$W_C = \frac{W_A}{n_A} + \frac{W_B}{n_B} = \frac{1}{2} W_A + W_B$$

- ▶ to je konzistentní!

Nápad 1.2: Poměrné hlasování, jak řešit?



- Zkusme hodnotit důležitost webů takto:

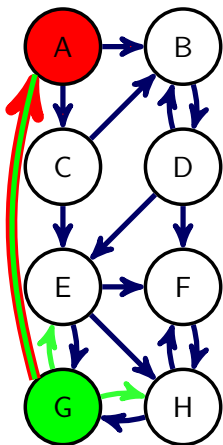
$$W_k = \sum_{b \in \{\rightarrow k\}} \frac{W_b}{n_b},$$

kde $\{\rightarrow k\}$ je množina webů, které odkazují na web k .

Notes

Neznámé máme v soustavě rovnic na obou stranách! Soustava je konzistentní, ale stále nevíme, jak vyřešit. Odložme ten bohlav na chvíli.

Nápad 1.2: Poměrné hlasování, jak řešit?



- Zkusme hodnotit důležitost webů takto:

$$W_k = \sum_{b \in \{\rightarrow k\}} \frac{W_b}{n_b},$$

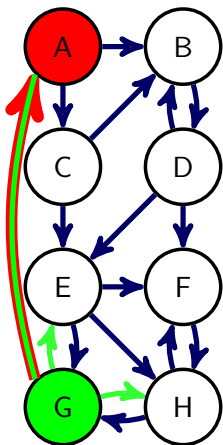
kde $\{\rightarrow k\}$ je množina webů, které odkazují na web k .

- Příklad: $W_A = \frac{W_G}{3}$

Notes

Neznáme máme v soustavě rovnic na obou stranách! Soustava je konzistentní, ale stále nevíme, jak vyřešit. Odložme ten bohlav na chvíli.

Nápad 1.2: Poměrné hlasování, jak řešit?



- Zkusme hodnotit důležitost webů takto:

$$W_k = \sum_{b \in \{\rightarrow k\}} \frac{W_b}{n_b},$$

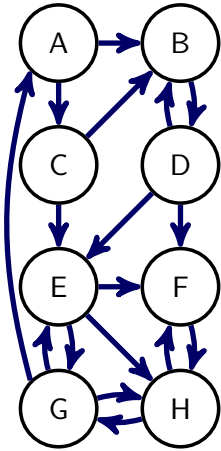
kde $\{\rightarrow k\}$ je množina webů, které odkazují na web k .

- Příklad: $W_A = \frac{W_G}{3}$
- Jak takovou soustavu rovnic vyřešit?
- Jde to vůbec? (pro obecný web)?
- Je řešení jen jedno, nebo víc?

Notes

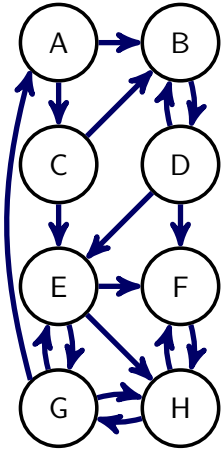
Neznáme máme v soustavě rovnic na obou stranách! Soustava je konzistentní, ale stále nevíme, jak vyřešit. Odložme ten bolehlav na chvílku.

Nápad 2: Náhodná procházka

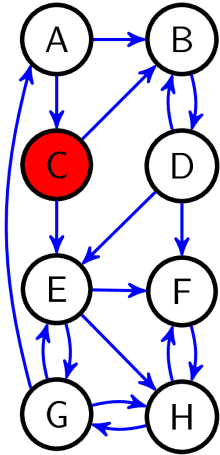


- ▶ co kdybychom nemuseli počítat skoro nic?
- ▶ máme strukturu webu k dispozici – jak bychom toho mohli využít
- ▶ nechme robota, aby procházel web následujícím způsobem:
 - ▶ začni na náhodné stránce
 - ▶ na této stránce je X odkazů. Náhodně se rozhodni pro jeden z nich
 - ▶ opakuj
 - ▶ počítej, kolikrát se dostaneš na jednotlivé stránky
- ▶ to celkem dává smysl, a vystihuje to konektivitu sítě

Nápad 2: Náhodná procházka

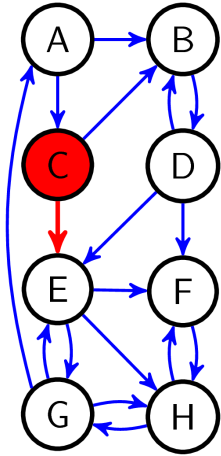


Nápad 2: Náhodná procházka



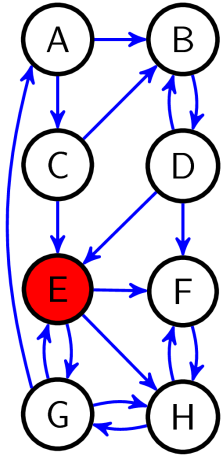
► začnu na C; 2 možná pokračování

Nápad 2: Náhodná procházka



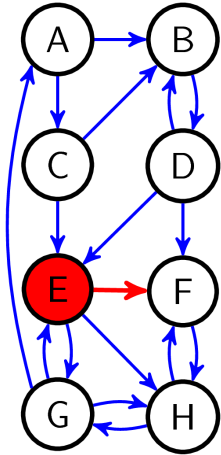
- ▶ začnu na C; 2 možná pokračování
- ▶ náhodně vyberu

Nápad 2: Náhodná procházka



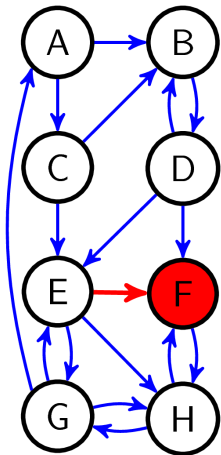
- ▶ začnu na C; 2 možná pokračování
- ▶ náhodně vyberu
- ▶ dostanu se do E; 3 cesty

Nápad 2: Náhodná procházka



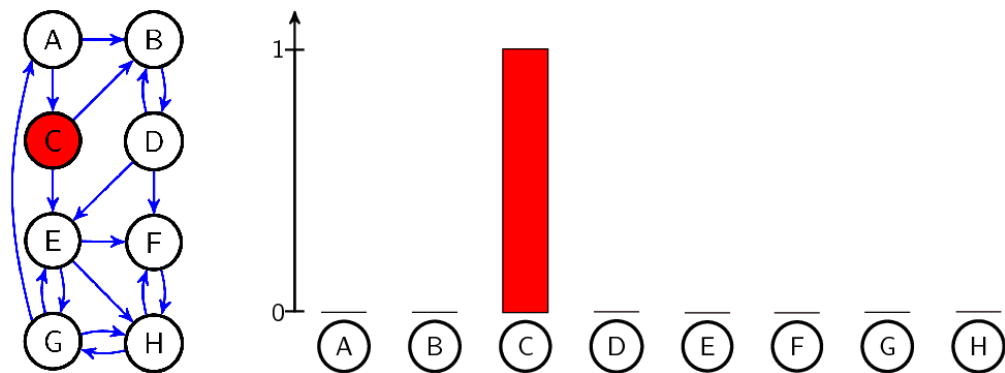
- ▶ začnu na C; 2 možná pokračování
- ▶ náhodně vyberu
- ▶ dostanu se do E; 3 cesty
- ▶ náhodně vyberu

Nápad 2: Náhodná procházka

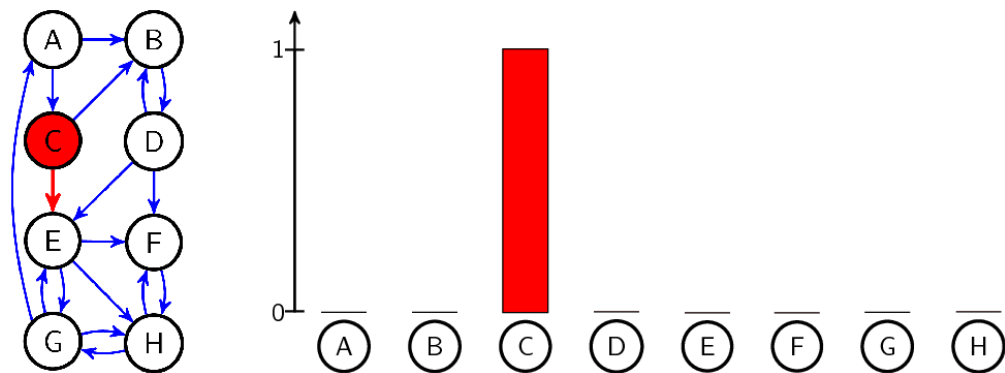


- ▶ začnu na C; 2 možná pokračování
- ▶ náhodně vyberu
- ▶ dostanu se do E; 3 cesty
- ▶ náhodně vyberu
- ▶ dostanu se do F
- ▶ ... a tak dále

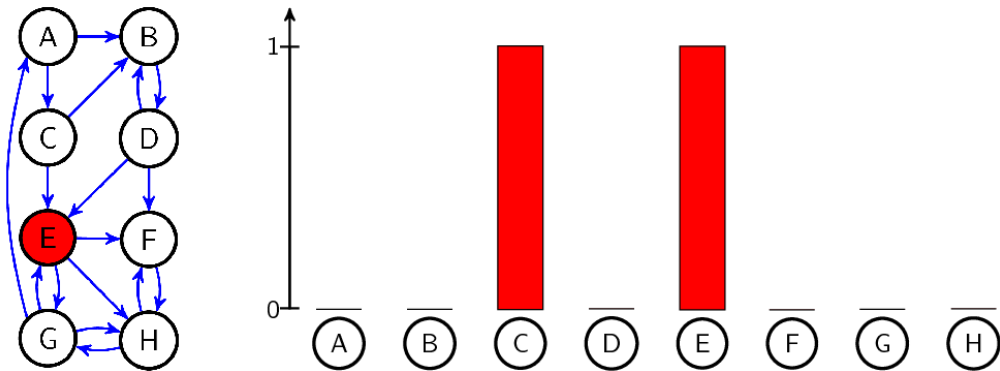
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



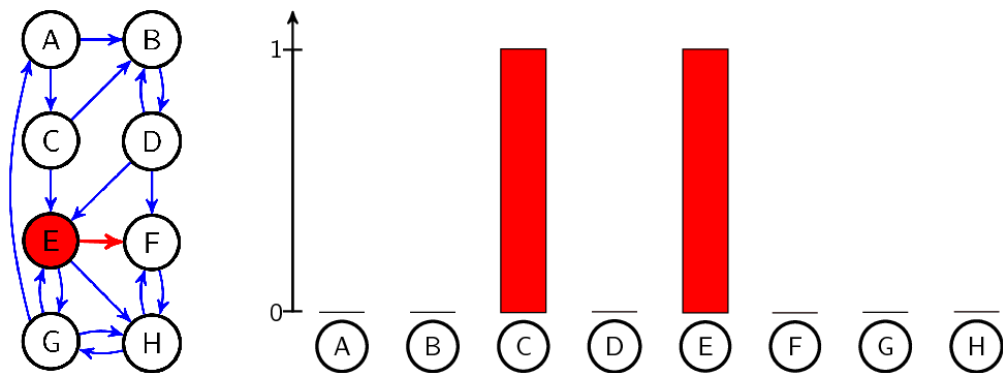
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



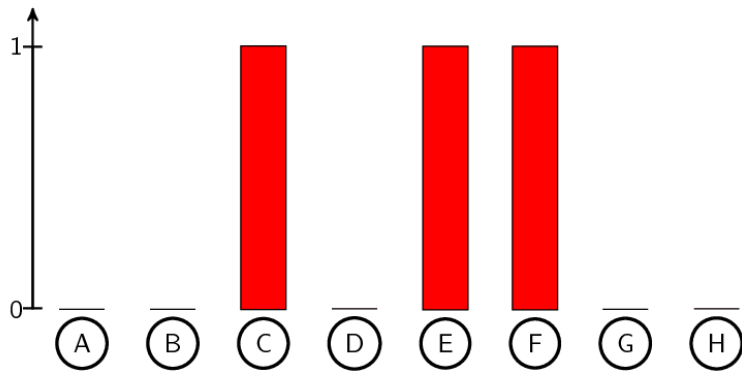
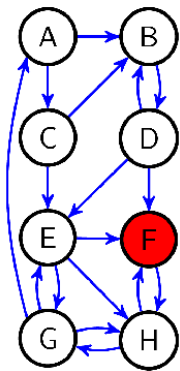
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



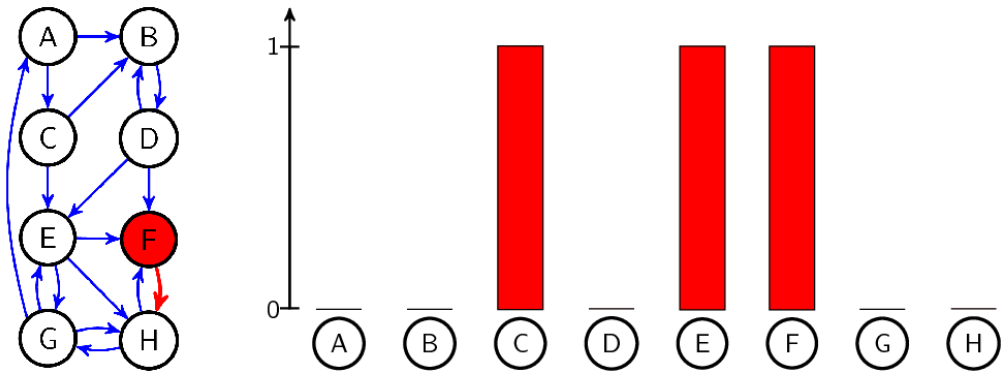
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



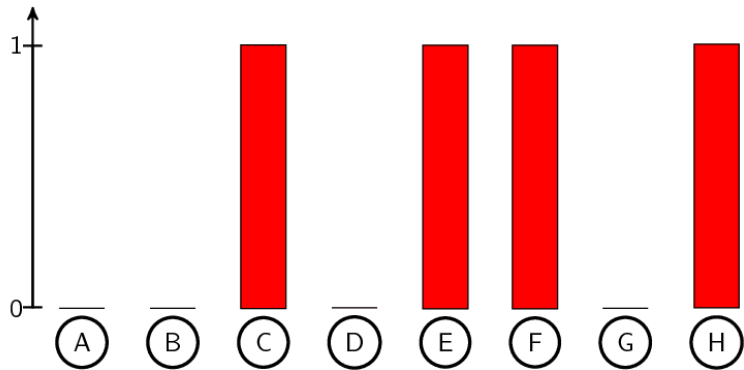
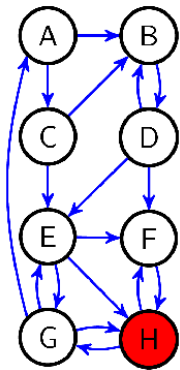
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



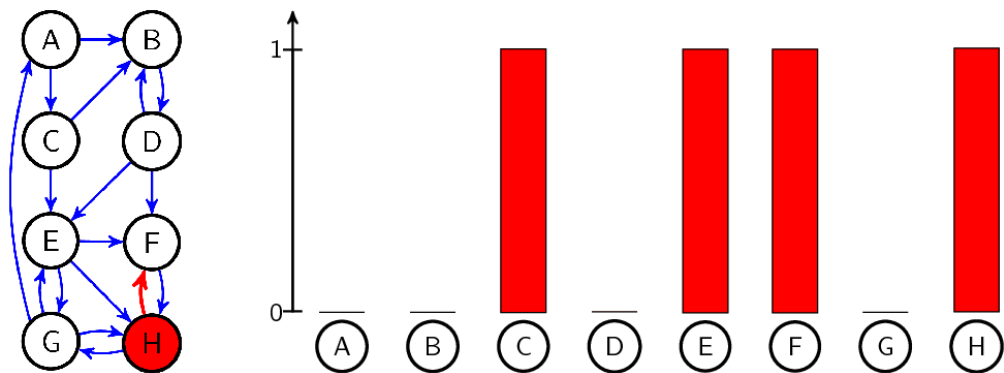
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



Nápad 2: Náhodná procházka, počítání průchodů (návštěv)

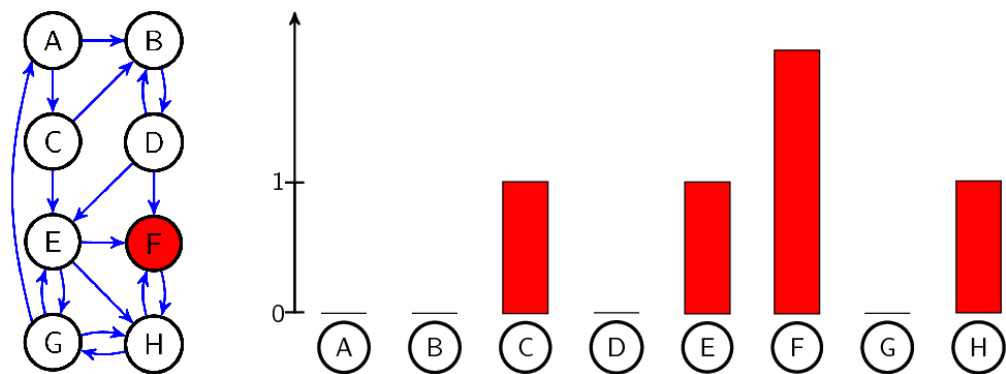


Nápad 2: Náhodná procházka, počítání průchodů (návštěv)

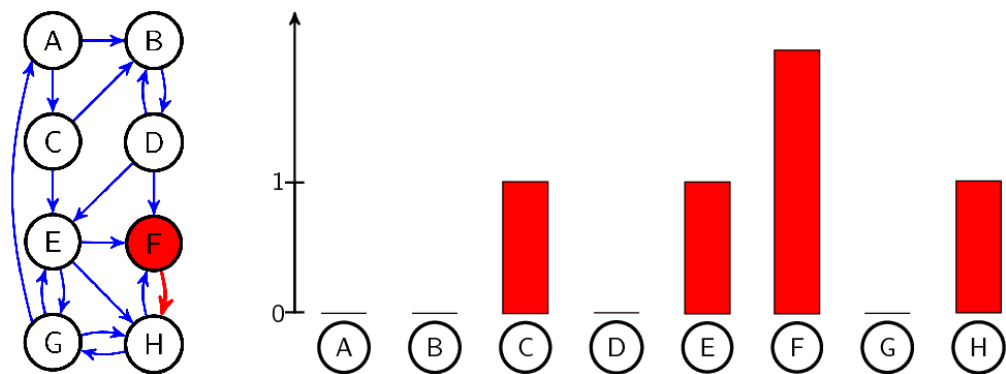


Notes

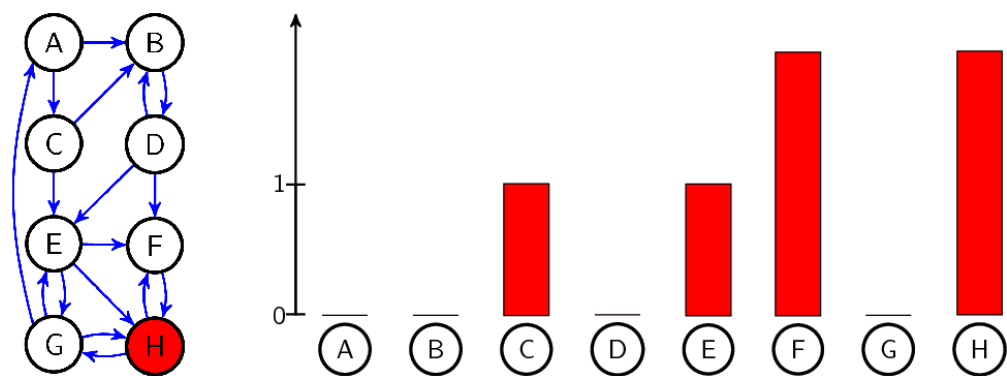
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



Nápad 2: Náhodná procházka, počítání průchodů (návštěv)

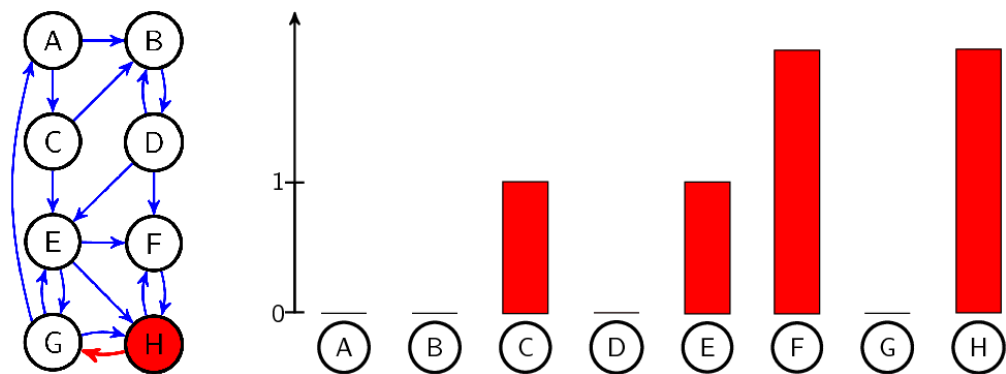


Nápad 2: Náhodná procházka, počítání průchodů (návštěv)

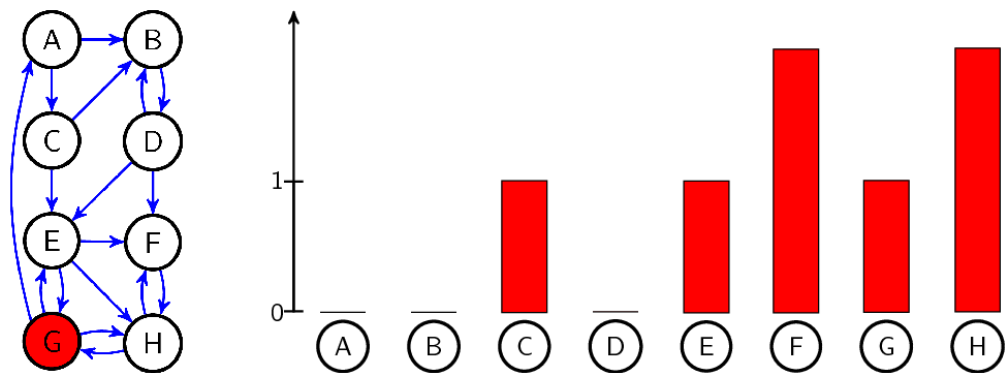


Notes

Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



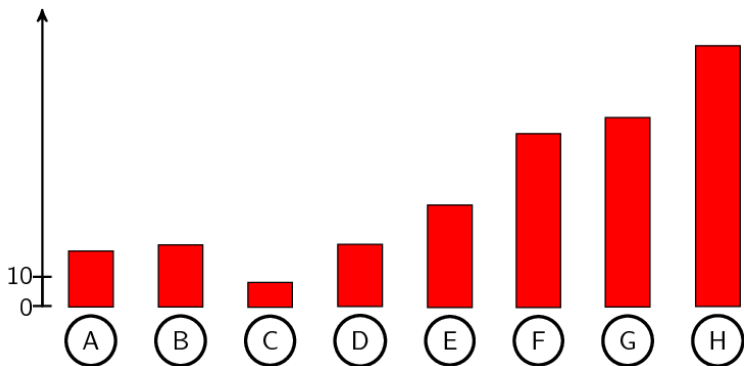
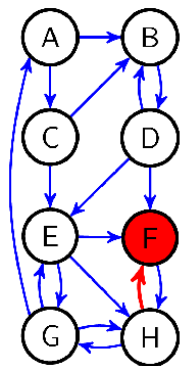
Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



Notes

Nápad 2: Náhodná procházka, počítání průchodů (návštěv)

300 iterací



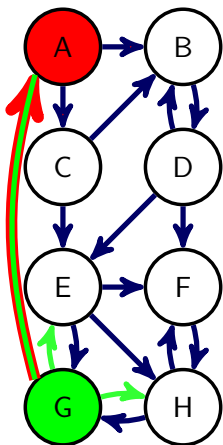
$T = (18, 20, 8, 20, 33, 56, 61, 84)$

Nápad 2: Náhodná procházka, počítání průchodů (návštěv)

$$T = (18, 20, 8, 20, 33, 56, 61, 84)$$

- ▶ dobře, z náhodné procházky dostaneme vždy počty průchodů
- ▶ co ta čísla jsou, je jasné. Mohu nějak vysvětlit, jak jednotlivá T souvisejí?

Nápad 2: Náhodná procházka, počítání průchodů (návštěv)



$$T = (18, 20, 8, 20, 33, 56, 61, 84)$$

- ▶ dobře, z náhodné procházky dostaneme vždy počty průchodů
- ▶ co ta čísla jsou, je jasné. Mohu nějak vysvětlit, jak jednotlivá T souvisejí?
- ▶ ano: pokud jsem například strávil na A čas T_A , mohlo se to stát jen proto, že jsem tam odněkud přišel, v tomto případě z G. Z G se chodí do A v $1/3$ případech.
- ▶ $T_A \approx \frac{T_G}{3}$
- ▶ hmmm, to je zajímavé!

Naše dva nápady: Ekvivalence!

Nápad 1: Poměrné hlasování

- ▶ odvozeno na základě velmi jednoduchých úvah
- ▶ máme podmínky na to, jak mají spolu váhy W_k jednotlivých webů souviset
- ▶ zatím nevíme, jak tyto podmínky splnit a tak najít řešení

Nápad 2: Náhodná procházka

- ▶ přirozená úvaha při absenci dalších znalostí
- ▶ dokážeme modelovat problém, získat časy strávené na jednotlivých stránkách T_k
- ▶ teprve po krátké analýze se ukázalo, že $T_k = W_k$ a tedy Nápad 2 řeší soustavu rovnic z Nápadu 1!

System $W_k = \sum A_{kb} W_b$

Chceme, aby platilo

$$W_k = \sum_{b \in \{\rightarrow k\}} \frac{W_b}{n_b}$$

System $W_k = \sum A_{kb} W_b$

Chceme, aby platilo

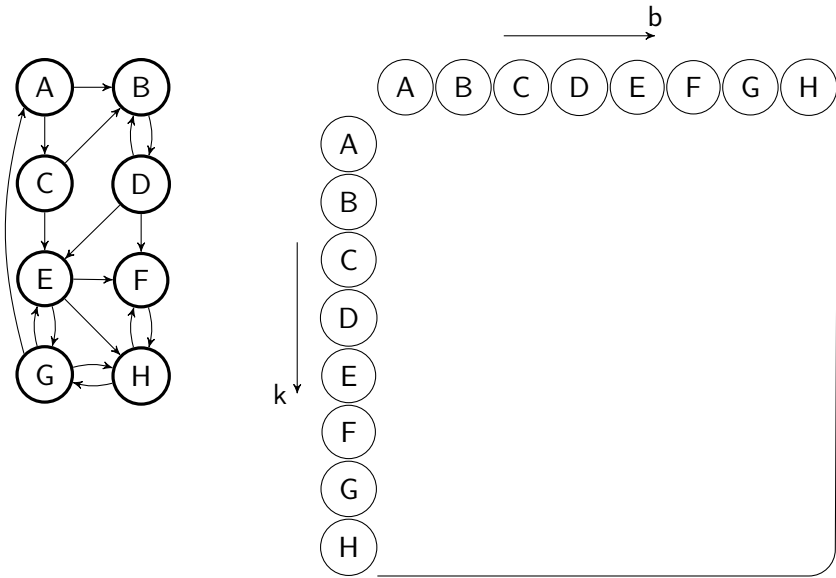
$$W_k = \sum_{b \in \{\rightarrow k\}} \frac{W_b}{n_b} = \sum_b A_{kb} W_b,$$

což je ekvivaletní maticovému zápisu

$$\mathbf{w} = \mathbf{A} \mathbf{w}.$$

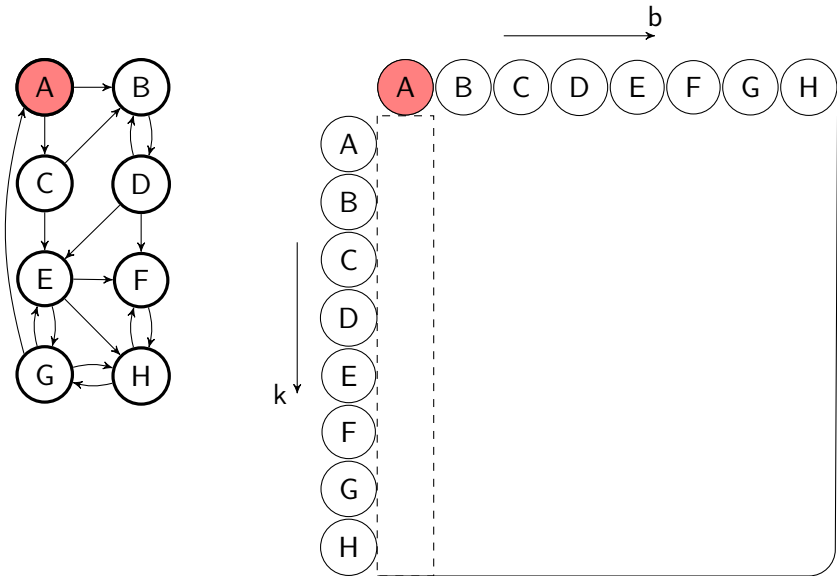
- ▶ A_{kb} je nenulové tam, kde existuje odkaz $b \rightarrow k$.
- ▶ je rovno $1/(\text{počet odkazů z } b)$

Konstrukce matice A, matice sousednosti (Adjacency)



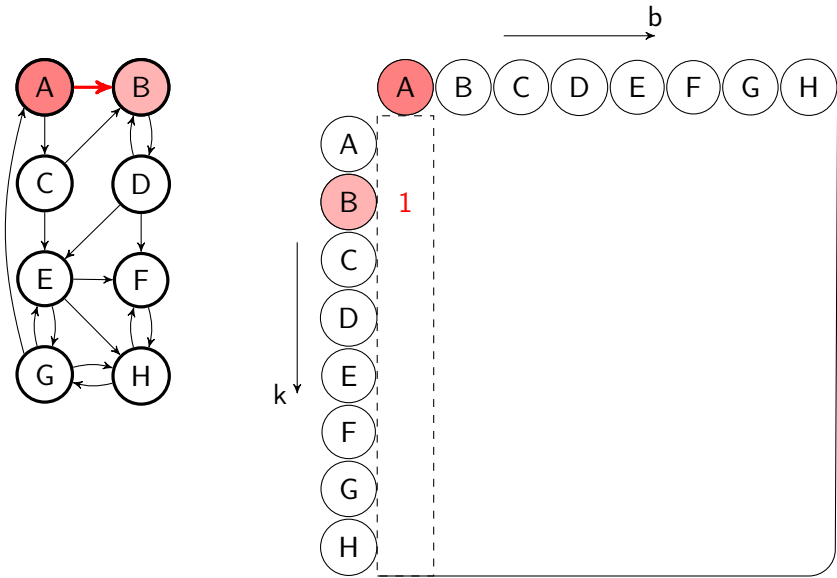
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



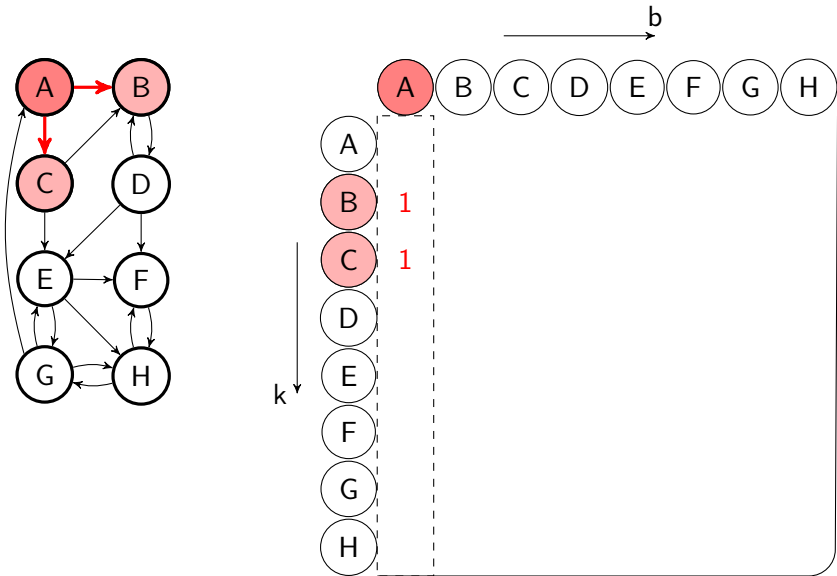
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



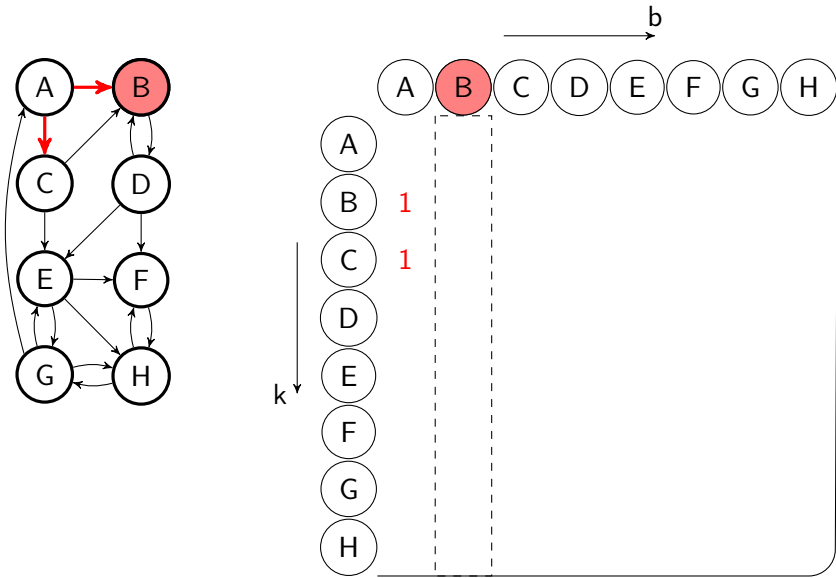
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



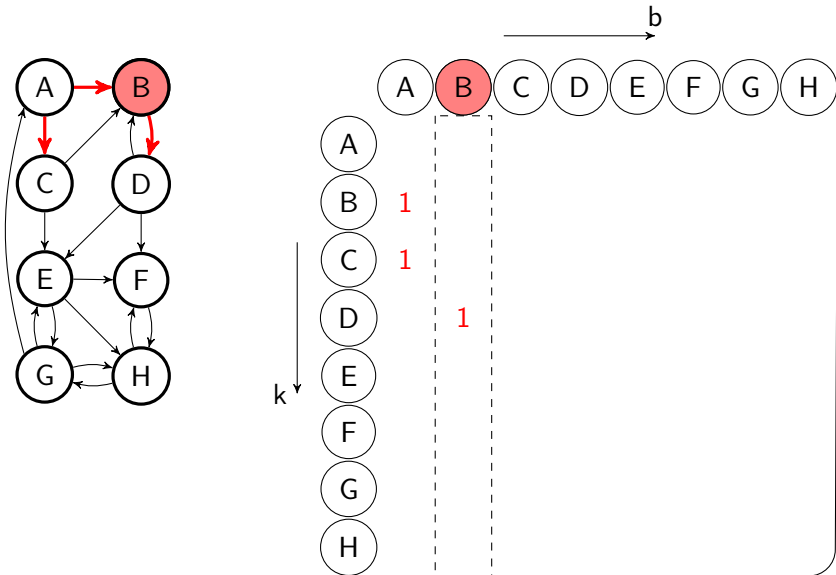
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



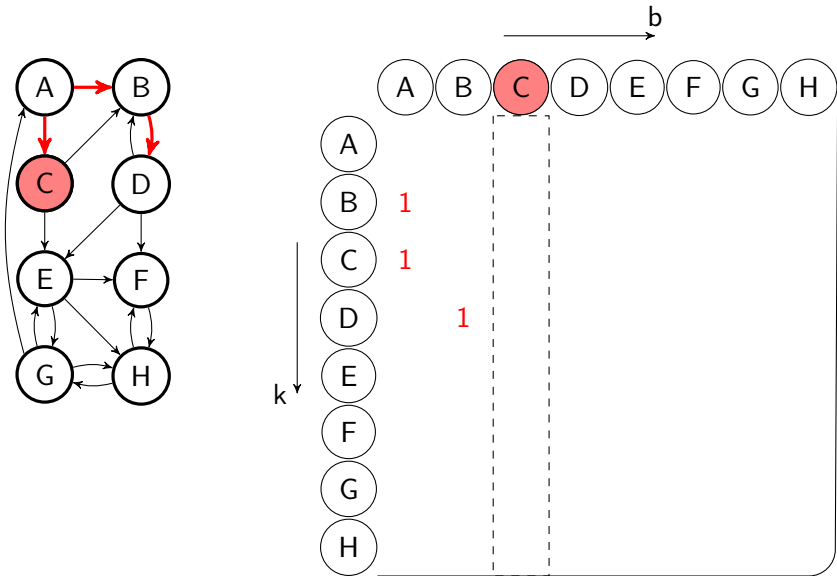
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



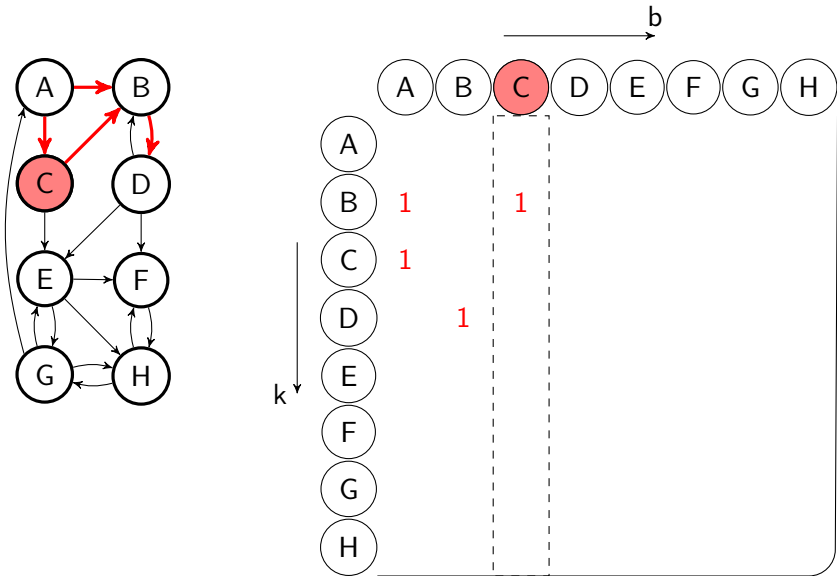
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



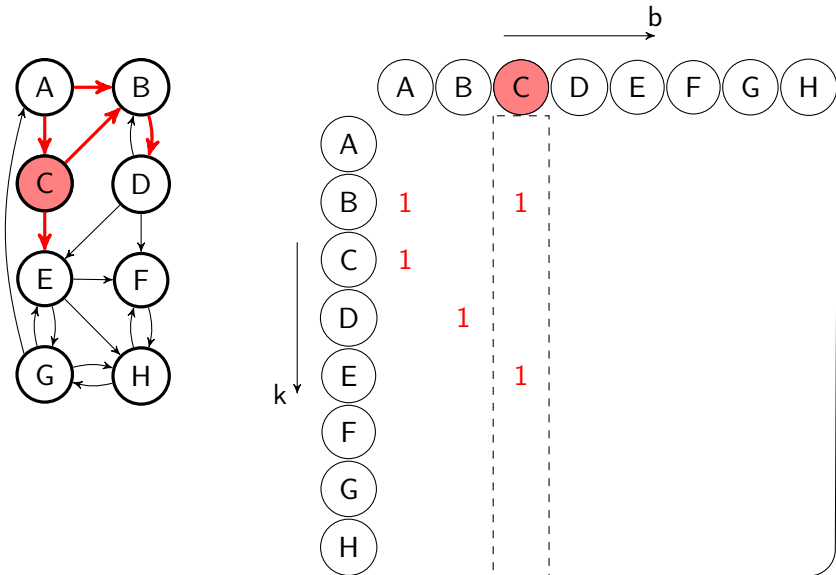
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



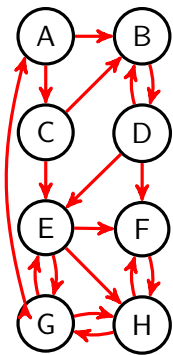
Notes

Konstrukce matice A, matice sousednosti (Adjacency)



Notes

Konstrukce matice A, matice sousednosti (Adjacency)

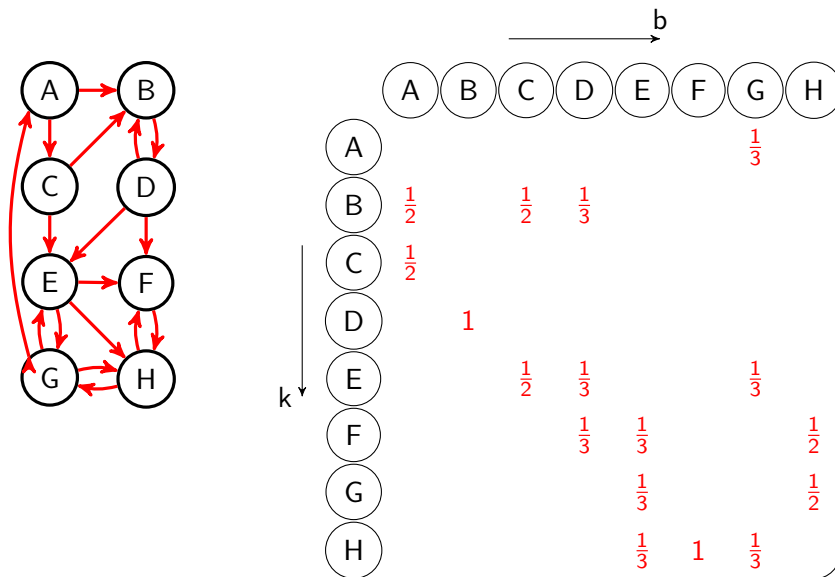


adjacency matrix
(matice sousednosti)

	b A B C D E F G H							
k A B C D E F G H								
	1		1	1				
	1							
		1						
			1	1			1	
				1	1			1
					1			1
					1	1	1	

Notes

Konstrukce matice A, normování



Notes

Čtvercové matici, kde je součet prvků v každém sloupci roven 1 se říká *stochastická* matice, říká je i matice přechodů,

System $W_k = \sum A_{kb} W_b$ ($\mathbf{w} = A\mathbf{w}$): řešení

$$\mathbf{w} = A\mathbf{w}$$

System $W_k = \sum A_{kb} W_b$ ($\mathbf{w} = \mathbf{A}\mathbf{w}$): řešení

$$\mathbf{w} = \mathbf{A}\mathbf{w}$$

► co je to za rovnici?

System $W_k = \sum A_{kb} W_b$ ($\mathbf{w} = \mathbf{A}\mathbf{w}$): řešení

$$\lambda \mathbf{w} = A \mathbf{w}$$

- ▶ co je to za rovnici?
- ▶ a ted'?

Systém $W_k = \sum A_{kb} W_b$ ($\mathbf{w} = A\mathbf{w}$): řešení

$$\lambda \mathbf{w} = A\mathbf{w}$$

- ▶ co je to za rovnici?
- ▶ a teď?
- ▶ ale vždyť to vypadá jako rovnice pro vlastní vektory

System $W_k = \sum A_{kb} W_b$ ($\mathbf{w} = \mathbf{A}\mathbf{w}$): řešení

$$\lambda \mathbf{w} = A\mathbf{w}$$

- ▶ co je to za rovnici?
- ▶ a teď?
- ▶ ale vždyť to vypadá jako rovnice pro vlastní vektory ($\lambda = 1$)!

System $W_k = \sum A_{kb} W_b$ ($\mathbf{w} = \mathbf{A}\mathbf{w}$): řešení

$$\lambda \mathbf{w} = A \mathbf{w}$$

- ▶ co je to za rovnici?
- ▶ a teď?
- ▶ ale vždyť to vypadá jako rovnice pro vlastní vektory ($\lambda = 1$)!
- ▶ existuje však vlastní vektor pro $\lambda = 1$?

$\lambda \mathbf{w} = \mathbf{Aw}$, existuje vlastní vektor pro $\lambda = 1$?

	A	B	C	D	E	F	G	H
A							$\frac{1}{3}$	
B	$\frac{1}{2}$							
C	$\frac{1}{2}$							
D		1						
E		$\frac{1}{2}$	$\frac{1}{3}$				$\frac{1}{3}$	
F			$\frac{1}{3}$	$\frac{1}{3}$				$\frac{1}{2}$
G				$\frac{1}{3}$				$\frac{1}{2}$
H				$\frac{1}{3}$	1	$\frac{1}{3}$		

Notes

Víme, že vlastní čísla nalezneme jako řešení rovnice

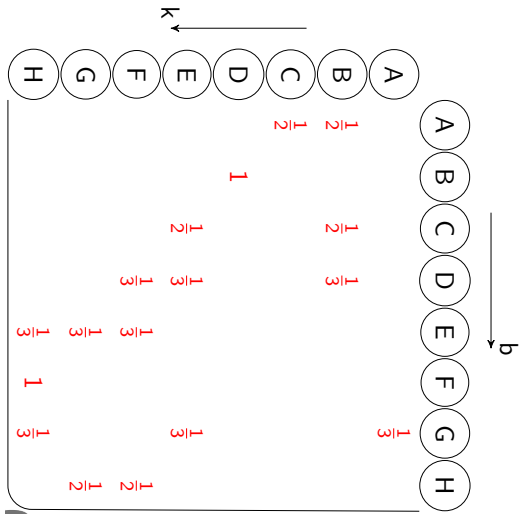
$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0$$

Determinant matice je stejný pro transponovanou matici.

Další diskuse např. :

<https://khanovaskola.cz/video/441/3655-determinant-transponovane-matice>

$\lambda \mathbf{w} = \mathbf{Aw}$, existuje vlastní vektor pro $\lambda = 1$?



Notes

Víme, že vlastní čísla nalezneme jako řešení rovnice

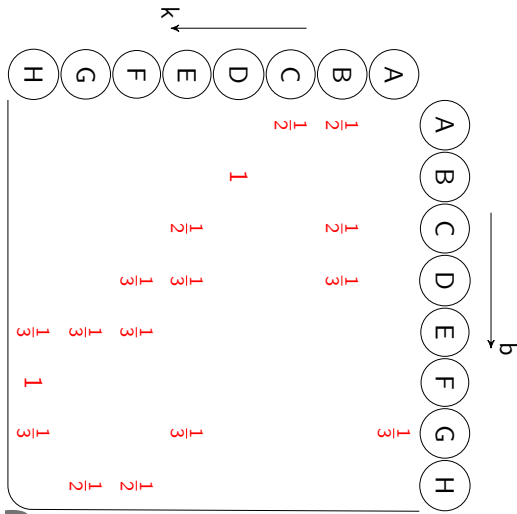
$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0$$

Determinant matice je stejný pro transponovanou matici.

Další diskuse např. :

<https://khanovaskola.cz/video/441/3655-determinant-transponovane-matice>

$\lambda \mathbf{w} = \mathbf{Aw}$, existuje vlastní vektor pro $\lambda = 1$?



$$\forall b: \sum_k A_{kb} = 1$$

Notes

Víme, že vlastní čísla nalezneme jako řešení rovnice

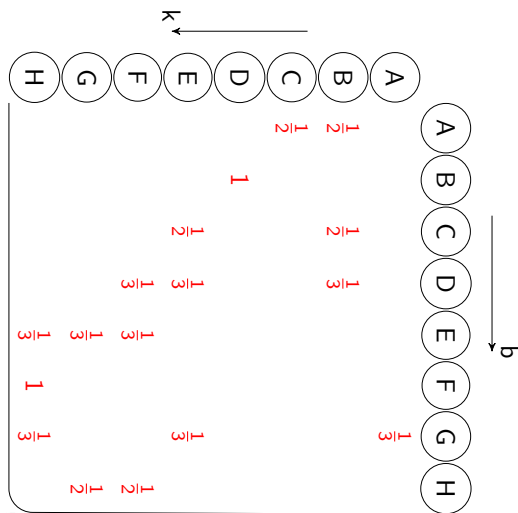
$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0$$

Determinant matice je stejný pro transponovanou matici.

Další diskuse např. :

<https://khanovaskola.cz/video/441/3655-determinant-transponovane-matice>

$\lambda \mathbf{w} = \mathbf{Aw}$, existuje vlastní vektor pro $\lambda = 1$?



$$\forall b: \sum_k A_{kb} = 1$$

to je však totéž, co

$$A^T \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

a vlastní čísla matic a transponovaných matic jsou stejná.

Notes

Víme, že vlastní čísla nalezneme jako řešení rovnice

$$\det(A - \lambda I) = 0$$

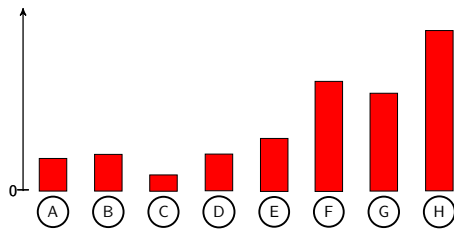
Determinant matice je stejný pro transponovanou matici.

Další diskuse např. :

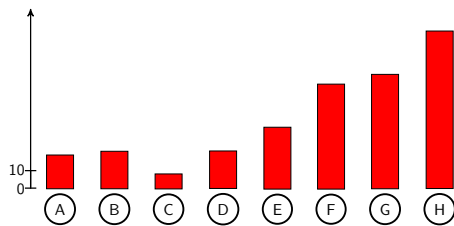
<https://khanovaskola.cz/video/441/3655-determinant-transponovane-matice>

Vlastní vektor $\mathbf{v}_1$

- ▶ vypočtený vektor $\mathbf{v}_1$



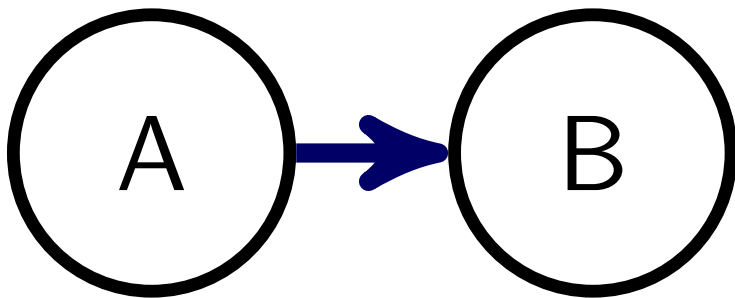
- ▶ srovnání s četností navštívení jednotlivých webů (náhodná procházka, 300 iterací)



Máme hotovo?

- Bude fungovat vždy?

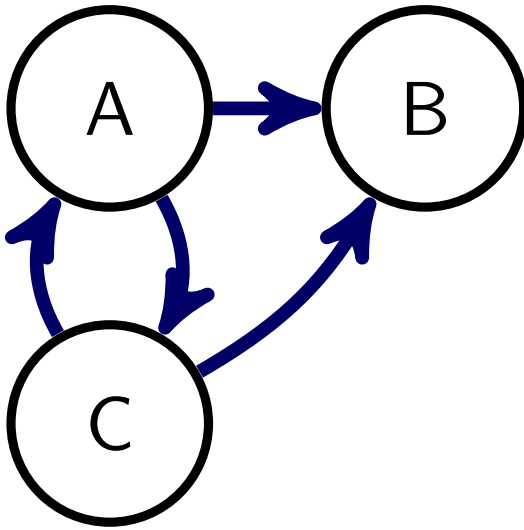
Problém 1 - dangling nodes



Notes

Zkuste si napsat matici sousednosti A nebo si odkrokujte náhodnou procházku.

Problém 1a - dangling (sink) nodes; Je něco špatně s A?



Notes

Zkuste si napsat matici sousednosti A nebo si odkrokujte náhodnou procházku.

$$G = A + D$$

D je nulová matice, s výjimkou sloupců, kde je v A nulový sloupec.

Notes

- V podstatě říkám, že se z izolovaného uzlu (stránky) mohou náhodně ocitnout v libovolném jiném uzlu.

$$G = A + D$$

D je nulová matice, s výjimkou sloupců, kde je v A nulový sloupec.

G je vždy stochastická. Máme už vyhráno?

Notes

- V podstatě říkám, že se z izolovaného uzlu (stránky) mohu náhodně ocitnout v libovolném jiném uzlu.

Jak se to ale spočítá?

A je velmi velká, n uzlů v našem webu, $\text{eig}(A)$ má složitost $\mathcal{O}(n^3)$.

Notes

Předpokládáme, že existuje báze vlastních vektorů matice A , $\mathbf{v}_j$. Vektor $\mathbf{v}^0$, resp. libovolný vektor $\mathbf{v}$ daném prostoru můžeme vyjádřit jako jejich lineární kombinaci; a váhy jsou souřadnice vektoru. Můžeme tedy začátek iterace napsat pomocí

$$\mathbf{v}^0 = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + c_3 \mathbf{v}_3 + \cdots + c_n \mathbf{v}_n.$$

Tedy $\mathbf{v}^0 = [c_1, c_2, c_3, \dots, c_n]^\top$. Pro vlastní vektory platí $A\mathbf{v}_j = \lambda_j \mathbf{v}_j$. Pak:

$$\begin{aligned} \mathbf{v}^1 &= \mathbf{A}\mathbf{v}^0 = c_1\mathbf{v}_1 + c_2\lambda_2\mathbf{v}_2 + c_3\lambda_3\mathbf{v}_3 + \cdots + c_n\lambda_n\mathbf{v}_n \\ \mathbf{v}^2 &= \mathbf{A}\mathbf{v}^1 = c_1\mathbf{v}_1 + c_2\lambda_2^2\mathbf{v}_2 + c_3\lambda_3^2\mathbf{v}_3 + \cdots + c_n\lambda_n^2\mathbf{v}_n \\ &\vdots \\ \mathbf{v}^k &= \mathbf{A}\mathbf{v}^{k-1} = c_1\mathbf{v}_1 + c_2\lambda_2^k\mathbf{v}_2 + c_3\lambda_3^k\mathbf{v}_3 + \cdots + c_n\lambda_n^k\mathbf{v}_n \end{aligned}$$

Připomeňme, že

$$1 = \lambda_1 > |\lambda_2| > |\lambda_3| > |\lambda_4| > \cdots > |\lambda_n|$$

Jak se to ale spočítá?

A je velmi velká, n uzlů v našem webu, $\text{eig}(A)$ má složitost $\mathcal{O}(n^3)$.
Naštěstí je také řídká!

01

Navigation icons and page number 36/43

Notes

Předpokládáme, že existuje báze vlastních vektorů matice A , $\mathbf{v}_j$. Vektor $\mathbf{v}^0$, resp. libovolný vektor v daném prostoru můžeme vyjádřit jako jejich lineární kombinaci; a váhy jsou souřadnice vektoru. Můžeme tedy začátek iterace napsat pomocí

$$\mathbf{v}^0 = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + c_3 \mathbf{v}_3 + \cdots + c_n \mathbf{v}_n.$$

Tedy $\mathbf{v}^0 = [c_1, c_2, c_3, \dots, c_n]^\top$. Pro vlastní vektory platí $A\mathbf{v}_j = \lambda_j \mathbf{v}_j$. Pak:

$$\begin{aligned} \mathbf{v}^1 &= A\mathbf{v}^0 = c_1 \lambda_1 \mathbf{v}_1 + c_2 \lambda_2 \mathbf{v}_2 + c_3 \lambda_3 \mathbf{v}_3 + \cdots + c_n \lambda_n \mathbf{v}_n \\ \mathbf{v}^2 &= A\mathbf{v}^1 = c_1 \lambda_1^2 \mathbf{v}_1 + c_2 \lambda_2^2 \mathbf{v}_2 + c_3 \lambda_3^2 \mathbf{v}_3 + \cdots + c_n \lambda_n^2 \mathbf{v}_n \\ &\vdots \\ \mathbf{v}^k &= A\mathbf{v}^{k-1} = c_1 \lambda_1^k \mathbf{v}_1 + c_2 \lambda_2^k \mathbf{v}_2 + c_3 \lambda_3^k \mathbf{v}_3 + \cdots + c_n \lambda_n^k \mathbf{v}_n \end{aligned}$$

Připomeňme, že

$$1 = \lambda_1 > |\lambda_2| > |\lambda_3| > |\lambda_4| > \cdots > |\lambda_n|$$

01

Naštěstí je také řídka!

- ▶ Metoda Power iteration:
 - ▶ začneme s náhodným jednotkovým vektorem $\mathbf{v}$
 - ▶ $\mathbf{v} \leftarrow \text{normalize}(\mathbf{A}\mathbf{v})$
 - ▶ iterujeme

Předpokládáme, že existuje báze vlastních vektorů matice A , $\mathbf{v}_j$. Vektor $\mathbf{v}^0$, resp. libovolný vektor $\mathbf{v}$ daném prostoru můžeme vyjádřit jako jejich lineární kombinaci; a váhy jsou souřadnice vektoru. Můžeme tedy začátek iterace napsat pomocí

$$\mathbf{v}^0 = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + c_3 \mathbf{v}_3 + \cdots + c_n \mathbf{v}_n.$$

Tedy $\mathbf{v}^0 = [c_1, c_2, c_3, \dots, c_n]^\top$. Pro vlastní vektory platí $A\mathbf{v}_j = \lambda_j\mathbf{v}_j$. Pak:

$$\begin{aligned} \mathbf{v}^1 &= \mathbf{A}\mathbf{v}^0 = c_1\mathbf{v}_1 + c_2\lambda_2\mathbf{v}_2 + c_3\lambda_3\mathbf{v}_3 + \cdots + c_n\lambda_n\mathbf{v}_n \\ \mathbf{v}^2 &= \mathbf{A}\mathbf{v}^1 = c_1\mathbf{v}_1 + c_2\lambda_2^2\mathbf{v}_2 + c_3\lambda_3^2\mathbf{v}_3 + \cdots + c_n\lambda_n^2\mathbf{v}_n \\ &\vdots \\ \mathbf{v}^k &= \mathbf{A}\mathbf{v}^{k-1} = c_1\mathbf{v}_1 + c_2\lambda_2^k\mathbf{v}_2 + c_3\lambda_3^k\mathbf{v}_3 + \cdots + c_n\lambda_n^k\mathbf{v}_n \end{aligned}$$

Připomeňme, že

$$1 = \lambda_1 > |\lambda_2| > |\lambda_3| > |\lambda_4| > \cdots > |\lambda_n|$$

Jak se to ale spočítá?

A je velmi velká, n uzlů v našem webu, $\text{eig}(A)$ má složitost $\mathcal{O}(n^3)$.

Naštěstí je také řídka!

- ▶ Metoda Power iteration:
 - ▶ začneme s náhodným jednotkovým vektorem $\mathbf{v}$
 - ▶ $\mathbf{v} \leftarrow \text{normalize}(\mathbf{A}\mathbf{v})$
 - ▶ iterujeme
- ▶ toto funguje, protože hledáme vektor příslušející největšímu vlastnímu číslu

$$1 = \lambda_1 > |\lambda_2| > |\lambda_3| > |\lambda_4| > \cdots > |\lambda_n|$$

- ▶ konvergence je tím rychlejší, čím má druhé největší vlastní číslo větší odstup od prvního
- ▶ velice jednoduchý algoritmus, efektivní pro obrovské a řídké matice

Notes

Předpokládáme, že existuje báze vlastních vektorů matice A , $\mathbf{v}_j$. Vektor $\mathbf{v}^0$, resp. libovolný vektor $\mathbf{v}$ daném prostoru můžeme vyjádřit jako jejich lineární kombinaci; a váhy jsou souřadnice vektoru. Můžeme tedy začátek iterace napsat pomocí

$$\mathbf{v}^0 = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + c_3 \mathbf{v}_3 + \cdots + c_n \mathbf{v}_n.$$

Tedy $\mathbf{v}^0 = [c_1, c_2, c_3, \dots, c_n]^\top$. Pro vlastní vektory platí $A\mathbf{v}_j = \lambda_j \mathbf{v}_j$. Pak:

$$\begin{aligned} \mathbf{v}^1 &= \mathbf{A}\mathbf{v}^0 = c_1\mathbf{v}_1 + c_2\lambda_2\mathbf{v}_2 + c_3\lambda_3\mathbf{v}_3 + \cdots + c_n\lambda_n\mathbf{v}_n \\ \mathbf{v}^2 &= \mathbf{A}\mathbf{v}^1 = c_1\mathbf{v}_1 + c_2\lambda_2^2\mathbf{v}_2 + c_3\lambda_3^2\mathbf{v}_3 + \cdots + c_n\lambda_n^2\mathbf{v}_n \\ &\vdots \\ \mathbf{v}^k &= \mathbf{A}\mathbf{v}^{k-1} = c_1\mathbf{v}_1 + c_2\lambda_2^k\mathbf{v}_2 + c_3\lambda_3^k\mathbf{v}_3 + \cdots + c_n\lambda_n^k\mathbf{v}_n \end{aligned}$$

Připomeňme, že

$$1 = \lambda_1 > |\lambda_2| > |\lambda_3| > |\lambda_4| > \cdots > |\lambda_n|$$

Power Iteration

$$\mathbf{w}^{k+1} = \mathbf{A}\mathbf{w}^k$$

Power Iteration

$$\mathbf{w}^{k+1} = \mathbf{A}\mathbf{w}^k$$

- Konverguje sekvence $\mathbf{w}^k$ vždy?

Power Iteration

$$\mathbf{w}^{k+1} = \mathbf{A}\mathbf{w}^k$$

- ▶ Konverguje sekvence $\mathbf{w}^k$ vždy?
- ▶ Je výsledný stacionární vektor nezávislý na $\mathbf{w}^0$?

Power Iteration

$$\mathbf{w}^{k+1} = \mathbf{A}\mathbf{w}^k$$

- Konverguje sekvence $\mathbf{w}^k$ vždy?
- Je výsledný stacionární vektor nezávislý na $\mathbf{w}^0$?
- Obsahuje výsledný stacionární vektor vždy to, co chcí?

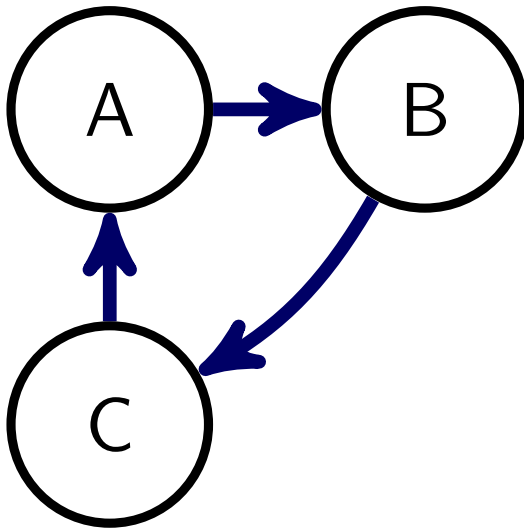
Power Iteration

$$\mathbf{w}^{k+1} = \mathbf{A}\mathbf{w}^k$$

- Konverguje sekvence $\mathbf{w}^k$ vždy?
- Je výsledný stacionární vektor nezávislý na $\mathbf{w}^0$?
- Obsahuje výsledný stacionární vektor vždy to, co chcí?

 $3 \times \text{Ne!}$

Problém 2 - Konvergence

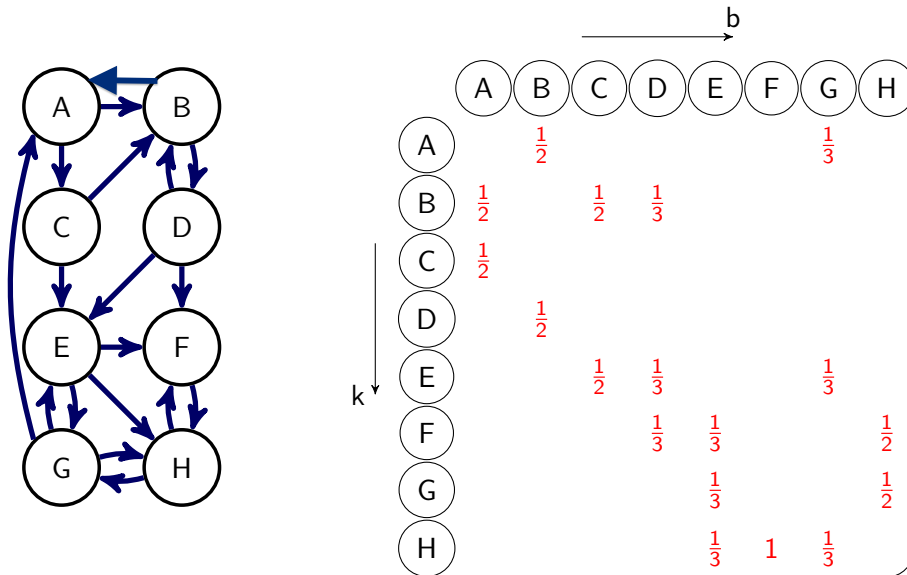


Notes

Vlastní vektor je ok, ale metoda power iter nekonverguje.

```
>> page_rank(cycle_matrix)
```

Problém 3 - Stacionární vektor



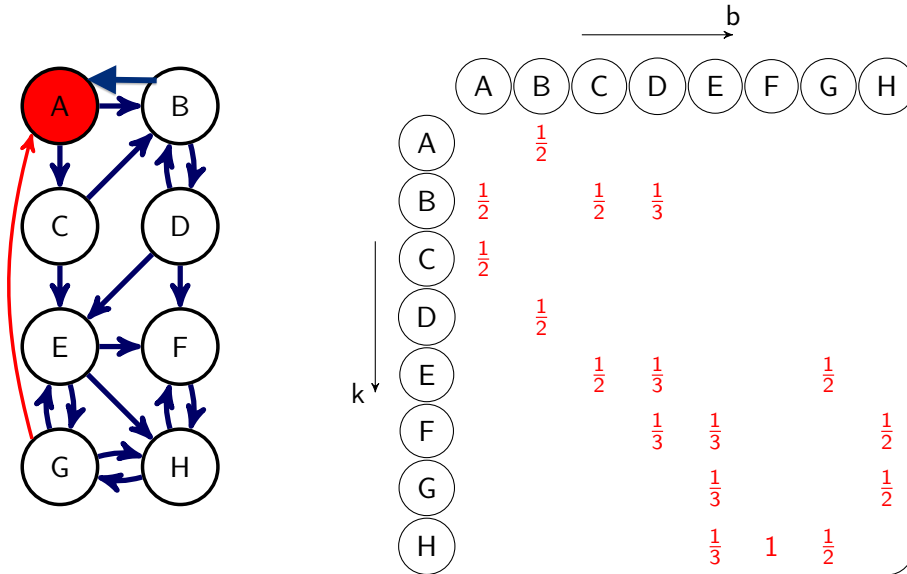
Notes

Zkonverguje, i vlastní vektor vypadá ok, ale je to hledané řešení?

```
>> page_rank(adjacency_matrix_deficient)
```

Matice A je rozložitelná (reducibilní/reducible)

Problém 3 - Stacionární vektor



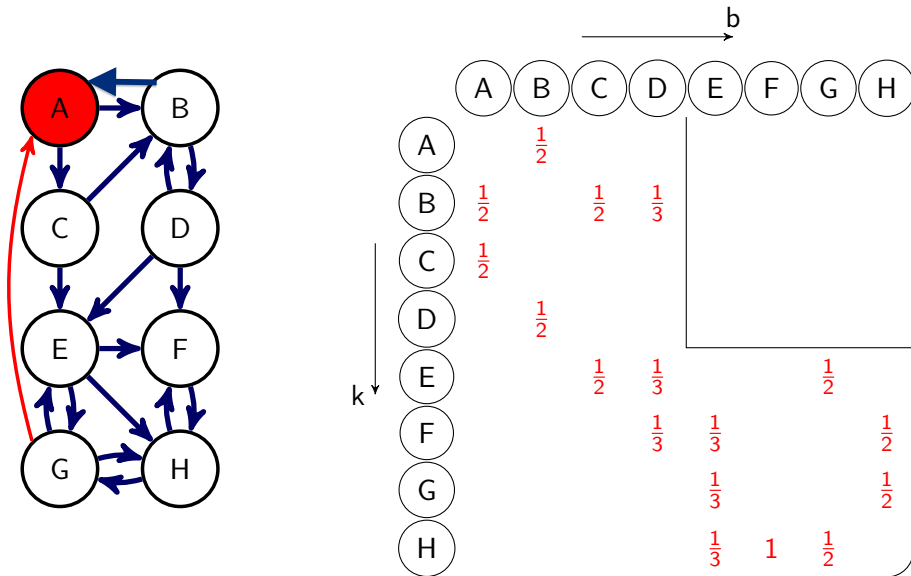
Notes

Zkonverguje, i vlastní vektor vypadá ok, ale je to hledané řešení?

```
>> page_rank(adjacency_matrix_deficient)
```

Matice A je rozložitelná (reducibilní/reducible)

Problém 3 - Stacionární vektor



Notes

Zkonverguje, i vlastní vektor vypadá ok, ale je to hledané řešení?

```
>> page_rank(adjacency_matrix_deficient)
```

Matice A je rozložitelná (reducibilní/reducible)

$$\mathbf{G} = \alpha(\mathbf{A} + \mathbf{D}) + (1 - \alpha)\frac{1}{n}\mathbf{I}$$

Notes

$$\mathbf{G}\mathbf{w} = \alpha\mathbf{A}\mathbf{w} + \alpha\mathbf{D}\mathbf{w} + (1 - \alpha)\frac{1}{n}\mathbf{I}\mathbf{w}$$

Matice $\mathbf{A}$ je řídká, to už víme. $\mathbf{D}$ má nulové a nenulové sloupce. Nenulové jsou všechny stejné! Při násobení jednotkovou maticí potřebujeme násobit jen jednou, resp. jedním řádkem matice $(1 - \alpha)\frac{1}{n}\mathbf{I}$. In literature¹, $\alpha = 0.85$ could be found.

¹David Austin: How Google Finds Your Needle in the Web's Haystack

$$\mathbf{G} = \alpha(\mathbf{A} + \mathbf{D}) + (1 - \alpha)\frac{1}{n}\mathbf{I}$$

G už není řádká, přesto je power iteration stále velmi efektivní. Proč?

01

Navigation icons and page number 40/43

Notes

$$\mathbf{G}\mathbf{w} = \alpha\mathbf{A}\mathbf{w} + \alpha\mathbf{D}\mathbf{w} + (1 - \alpha)\frac{1}{n}\mathbf{I}\mathbf{w}$$

Matice $\mathbf{A}$ je řádká, to už víme. $\mathbf{D}$ má nulové a nenulové sloupce. Nenulové jsou všechny stejné! Při násobení jednotkovou maticí potřebujeme násobit jen jednou, resp. jedním řádkem matice $(1 - \alpha)\frac{1}{n}\mathbf{I}$. In literature², $\alpha = 0.85$ could be found.

²David Austin: How Google Finds Your Needle in the Web's Haystack

Co plyne z odvození PageRanku?

- ▶ úžasný efekt mají mnohonásobné pohledy na věc. Jedno může pohánět druhé, analýza mohutně profituje.
- ▶ jednoduchá myšlenka, přístupná každému, která změnila svět
- ▶ neznamená to ale, že na ni každý přijde (známý efekt „no jistě!“, který se dostavuje po odhalení řešení). Dnes nám přijde velice přirozené, že všechny souřadné soustavy by měly být rovnocenné (Einsteinova teorie relativity)
- ▶ jednoduché věci (většinou) fungují
- ▶ je to potvrzení toho, že je třeba analyzovat problém postupně a postupovat od principiálně jednoduššího řešení ke složitějšímu

Google story

- ▶ 1995: Larry Page a Sergey Brin se potkali na Stanfordu (bylo jim kolem 21 let)
- ▶ 1996-1997: odvození, implementace hledacího stroje, vymyšlení názvu Google
- ▶ 1998: 'lepší než cokoli jiného'
- ▶ 2004: IPO na Wall St, \$85/akcie (market cap \$25 miliard)
- ▶ od počátku skutečný leader v inovacích

Reference

- ▶ wikipedia
- ▶ David Austin: How Google Finds Your Needle in the Web's Haystack
- ▶ KURT BRYAN AND TANYA LEISE: The \$25,000,000,000 Eigenvector: The Linear Algebra Behind Google