Generování matic bez zakázaných vzorů

Stanislav Kučera

Informatický ústav Univerzity Karlovy

8. 9. 2016

Zadání

Cílem bakálářské práce bylo navrhnout a implementovat postup, jak vytvořit aproximaci rovnoměrně náhodné binární matice neobsahující daný zakázaný vzor.

Zakázaný vzor

Definice

Binární matice $M \in \{0,1\}^{m \times n}$ obsahuje binární matici $P \in \{0,1\}^{k \times l}$ jako podmatici, pokud lze z M vynecháním některých řádků a sloupečků získat matici M' velikosti $k \times l$ takovou, že pokud má P jedničku na nějaké pozici, má na téže pozici jedničku i M'. Jinak řekneme, že M *neobsahuje (vyhýbá se)* P *jako podmatici*.

$$P = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad M_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \quad M_2 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

Zakázaný vzor

Definice

Binární matice $M \in \{0,1\}^{m \times n}$ obsahuje binární matici $P \in \{0,1\}^{k \times l}$ jako podmatici, pokud lze z M vynecháním některých řádků a sloupečků získat matici M' velikosti $k \times l$ takovou, že pokud má P jedničku na nějaké pozici, má na téže pozici jedničku i M'. Jinak řekneme, že M *neobsahuje (vyhýbá se)* P *jako podmatici*.

$$P = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad M_1 = \begin{pmatrix} 1 & 1 & 0 \\ \cancel{0} & \cancel{0} & \cancel{1} \\ 1 & 1 & 0 \end{pmatrix} \quad M_2 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

Příklady použití

- Za pomoci matic bez zakázaných vzorů se dokázal horní odhad časové složitosti algoritmu Efrata a Sharira na "Segment-center problem".
- Existuje korelace mezi některými třídami matic bez zakázených vzorů a Davenport-Schinzelovým posloupnostmi, které souvisí se složitostí dolní (horní) obálky arrangementů v rovině.

Markovovy řetězce

Definice (neformální)

Pro předepsané pravděpodobnosti $p_{i,j}$ je Markovův řetězec posloupnost $X_0, X_1, \ldots$ prvků ze stavové množiny $\mathcal{X}$ dodržující $P[X_{t+1} = j | X_t = i] = p_{i,j}$.

Markovovy řetězce

Definice (neformální)

Pro předepsané pravděpodobnosti $p_{i,j}$ je Markovův řetězec posloupnost $X_0, X_1, \ldots$ prvků ze stavové množiny $\mathcal{X}$ dodržující $P[X_{t+1} = j | X_t = i] = p_{i,j}$.

Věta (neformální)

Pokud je Markovův řetězec aperiodický, nerozložitelný a symetrický, potom je jeho limita uniformně náhodně rozložena na stavové množině $\mathcal{X}$.

Markovův řetězec pro matice

Pokud chceme generovat matici neobsahující vzor P, postupujeme takto:

1. Zvolíme libovolnou matici M neobahující P.
2. Změníme uniformně náhodně vybraný bit M, čímž dostaneme M'.
3. Pokud M' neobsahuje P jako podmatici, nastavíme $M := M'$.
4. Goto 2.

Markovův řetězec pro matice

Pokud chceme generovat matici neobsahující vzor P, postupujeme takto:

1. Zvolíme libovolnou matici M neobahující P.
2. Změníme uniformně náhodně vybraný bit M, čímž dostaneme M'.
3. Pokud M' neobsahuje P jako podmatici, nastavíme $M := M'$.
4. Goto 2.

Protože definovaný Markovův řetězec splňuje předpoklady věty z minulého slidu, je jeho limita náhodná matice neobsahující vzor P jako podmatici. My nemáme čas čekat nekonečně dlouho, a zároveň zmíněná věta ani žádná jiná (pro obecné Markovovy řetězce) nedává odhad na dostačující počet iterací (mixing time), proto volbu počtu iterací necháme na uživateli.

Algoritmus pro testování speciálních vzorů

Definice

O matici řekneme, že je to Walking pattern, pokud existuje procházka z levého horního rohu matice do pravého dolního rohu obsahující všechny jedničky v matici.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Algoritmus pro testování speciálních vzorů

Definice

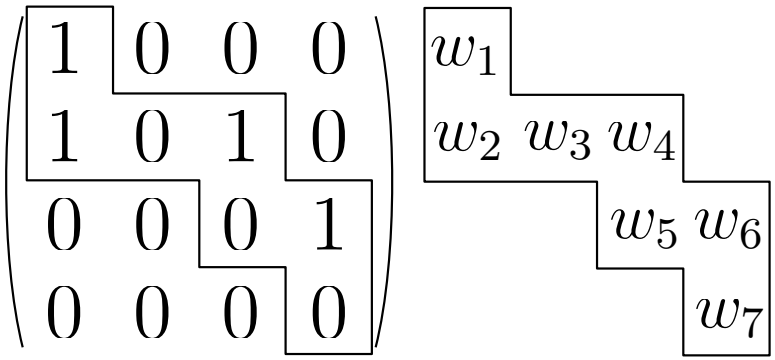
O matici řekneme, že je to Walking pattern, pokud existuje procházka z levého horního rohu matice do pravého dolního rohu obsahující všechny jedničky v matici.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Algoritmus pro testování speciálních vzorů

Definice

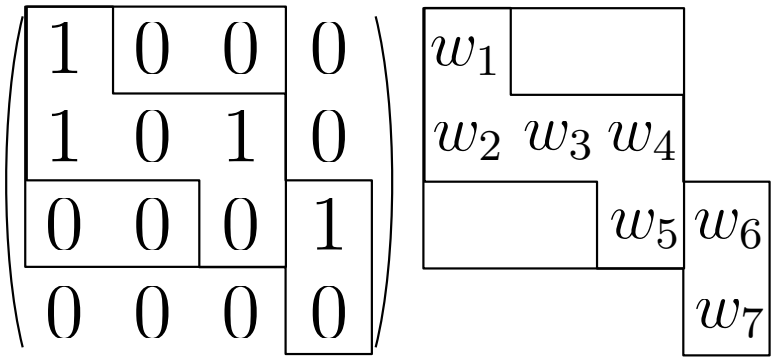
O matici řekneme, že je to Walking pattern, pokud existuje procházka z levého horního rohu matice do pravého dolního rohu obsahující všechny jedničky v matici.



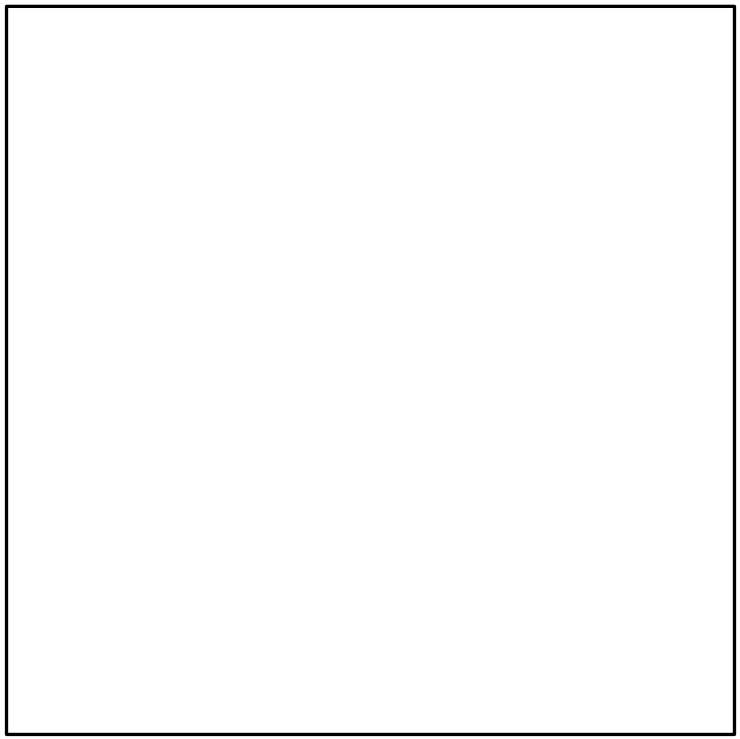
Algoritmus pro testování speciálních vzorů

Definice

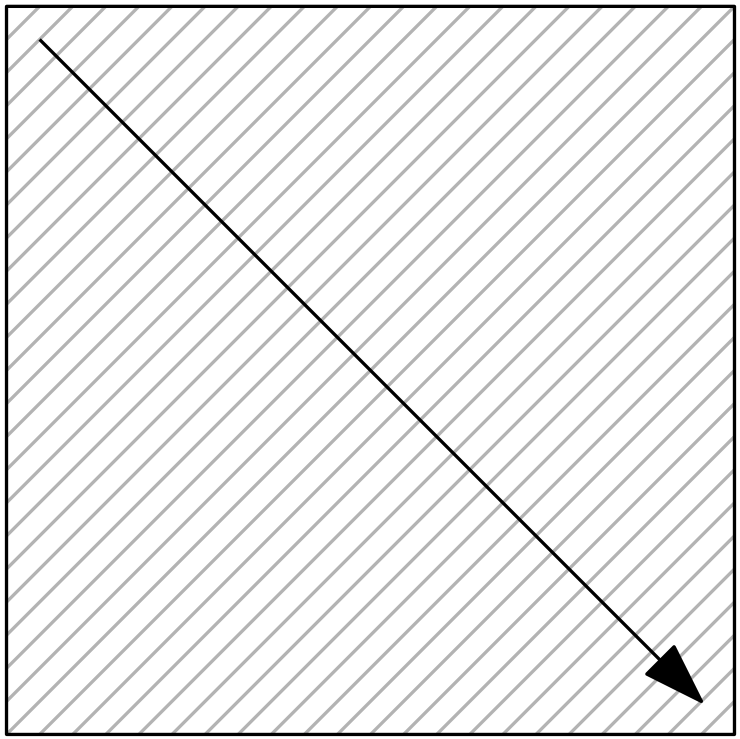
O matici řekneme, že je to Walking pattern, pokud existuje procházka z levého horního rohu matice do pravého dolního rohu obsahující všechny jedničky v matici.



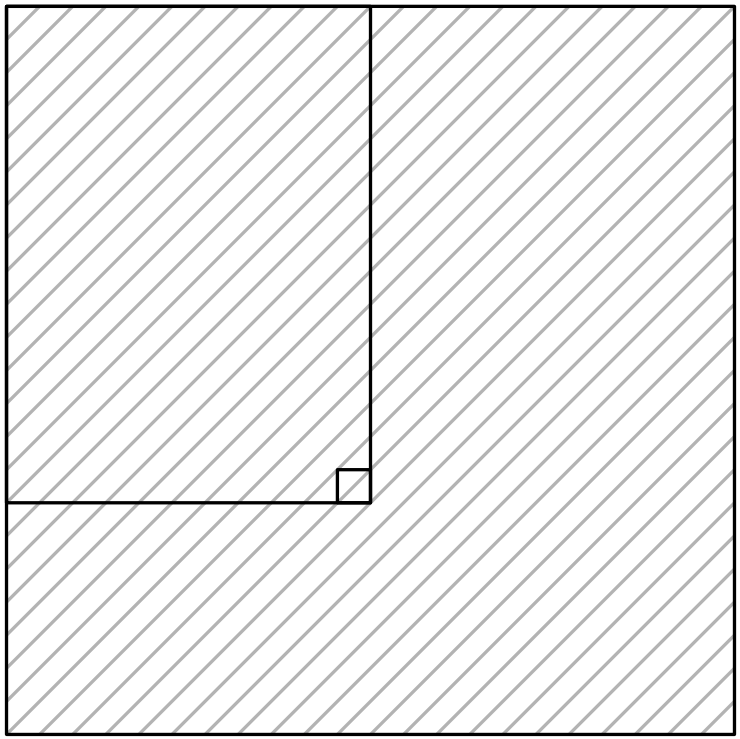
Algoritmus pro testování Walking pattern



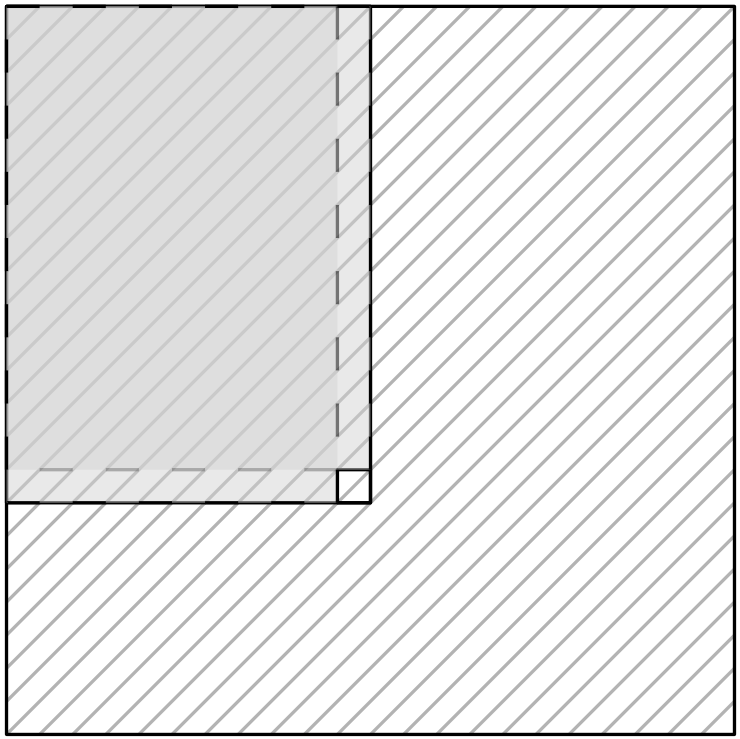
Algoritmus pro testování Walking pattern



Algoritmus pro testování Walking pattern



Algoritmus pro testování Walking pattern



Algoritmus pro testování obecných vzorů

Rozhodnout, zda daná matice obsahuje daný vzor je NP-úplné (dokonce i pro permutační matice).

Algoritmus pro testování obecných vzorů

Rozhodnout, zda daná matice obsahuje daný vzor je NP-úplné (dokonce i pro permutační matice).

Při testování obsahování vzoru postupně mapujeme všechny linie (řádky a sloupce) vzoru na všechny možné linie testované matice.

Algoritmus pro testování obecných vzorů

Rozhodnout, zda daná matice obsahuje daný vzor je NP-úplné (dokonce i pro permutační matice).

Při testování obsahování vzoru postupně mapujeme všechny linie (řádky a sloupce) vzoru na všechny možné linie testované matice.

Optimalizace:

- Některá částečná mapování můžeme sloučit a tím ušetřit čas i prostor.

Algoritmus pro testování obecných vzorů

Rozhodnout, zda daná matice obsahuje daný vzor je NP-úplné (dokonce i pro permutační matice).

Při testování obsahování vzoru postupně mapujeme všechny linie (řádky a sloupce) vzoru na všechny možné linie testované matice.

Optimalizace:

- Některá částečná mapování můžeme sloučit a tím ušetřit čas i prostor.
- Program poskytuje mnoho různých způsobů jak zvolit pořadí, ve kterém se budou linie mapovat.

Algoritmus pro testování obecných vzorů

Rozhodnout, zda daná matice obsahuje daný vzor je NP-úplné (dokonce i pro permutační matice).

Při testování obsahování vzoru postupně mapujeme všechny linie (řádky a sloupce) vzoru na všechny možné linie testované matice.

Optimalizace:

- Některá částečná mapování můžeme sloučit a tím ušetřit čas i prostor.
- Program poskytuje mnoho různých způsobů jak zvolit pořadí, ve kterém se budou linie mapovat.
- Volitelně program při mapování linie testuje, jestli je dost jedniček tam, kam se budou později mapovat dosud nenamapované linie.

Algoritmus pro testování obecných vzorů

Rozhodnout, zda daná matice obsahuje daný vzor je NP-úplné (dokonce i pro permutační matice).

Při testování obsahování vzoru postupně mapujeme všechny linie (řádky a sloupce) vzoru na všechny možné linie testované matice.

Optimalizace:

- Některá částečná mapování můžeme sloučit a tím ušetřit čas i prostor.
- Program poskytuje mnoho různých způsobů jak zvolit pořadí, ve kterém se budou linie mapovat.
- Volitelně program při mapování linie testuje, jestli je dost jedniček tam, kam se budou později mapovat dosud nenamapované linie.
- Protože známe generující proces a již víme, že matice před změnou bitu vzor neobsahovala, víme také, že pokud ho po změně obsahuje, tak jedině proto, že právě změněný bit je součástí mapování vzoru.

Vzor	n	#iterací	pořadí	one	rek	ort	čas (s)
	100	100 000	MAX	ano	ano	ano	121,85
	100	100 000	MAX	ano	ano	ne	120,80
	100	100 000	MAX	ne	ne	ne	263,71
	500	10 000	MAX	ano	ano	ano	1 053,18
1 0 0	500	10 000	MAX	ano	ano	ne	1 051,97
1 1 1	500	10 000	MAX	ne	ne	ne	2 695,26
0 0 1	100	100 000	DESC	ano	ano	ano	82,39
	100	100 000	DESC	ano	ano	ne	92,72
	100	100 000	DESC	ne	ne	ne	113,34
	500	10 000	DESC	ano	ano	ano	430,01
	500	10 000	DESC	ano	ano	ne	446,21
	500	10 000	DESC	ne	ne	ne	195,15

Vzor	n	#iterací	pořadí	one	rek	ort	čas (s)
	100	100 000	MAX	ano	ano	ano	121,85
	100	100 000	MAX	ano	ano	ne	120,80
	100	100 000	MAX	ne	ne	ne	263,71
	500	10 000	MAX	ano	ano	ano	1 053,18
1 0 0	500	10 000	MAX	ano	ano	ne	1 051,97
1 1 1	500	10 000	MAX	ne	ne	ne	2 695,26
0 0 1	100	100 000	DESC	ano	ano	ano	82,39
	100	100 000	DESC	ano	ano	ne	92,72
	100	100 000	DESC	ne	ne	ne	113,34
	500	10 000	DESC	ano	ano	ano	430,01
	500	10 000	DESC	ano	ano	ne	446,21
	500	10 000	DESC	ne	ne	ne	195,15

Vzor	n	#iterací	pořadí	one	rek	ort	čas (s)
	100	100 000	MAX	ano	ano	ano	121,85
	100	100 000	MAX	ano	ano	ne	120,80
	100	100 000	MAX	ne	ne	ne	263,71
	500	10 000	MAX	ano	ano	ano	1 053,18
1 0 0	500	10 000	MAX	ano	ano	ne	1 051,97
1 1 1	500	10 000	MAX	ne	ne	ne	2 695,26
0 0 1	100	100 000	DESC	ano	ano	ano	82,39
	100	100 000	DESC	ano	ano	ne	92,72
	100	100 000	DESC	ne	ne	ne	113,34
	500	10 000	DESC	ano	ano	ano	430,01
	500	10 000	DESC	ano	ano	ne	446,21
	500	10 000	DESC	ne	ne	ne	195,15

Vícevláknové generování

Pokud generujeme dostatečně velkou matici a už je dostatečně zahuštěná, potom pravděpodobnost, že nějaká změna bitu uspěje, je malá, a tedy většina iterací neuspěje (matice zůstane taková, jaká byla před iterací).

Vícevláknové generování

Pokud generujeme dostatečně velkou matici a už je dostatečně zahuštěná, potom pravděpodobnost, že nějaká změna bitu uspěje, je malá, a tedy většina iterací neuspěje (matice zůstane taková, jaká byla před iterací).

Idea: pro generovanou matici M si místo jedné vybere p pozic, pro každou samostatně změníme určených bit M, čímž dostaneme p matic M'.
Pro každou z nich otestujeme obsahování vzoru, a podle výsledků vybereme, které změny na matici M provedeme.

Vícevláknové generování

Pokud generujeme dostatečně velkou matici a už je dostatečně zahuštěná, potom pravděpodobnost, že nějaká změna bitu uspěje, je malá, a tedy většina iterací neuspěje (matice zůstane taková, jaká byla před iterací).

Idea: pro generovanou matici M si místo jedné vybere p pozic, pro každou samostatně změníme určených bit M, čímž dostaneme p matic M'. Pro každou z nich otestujeme obsahování vzoru, a podle výsledků vybereme, které změny na matici M provedeme.

Výběr změn, které ovlivní generovanou matici musíme dělat opatrně, aby stále neobsahovala zakázaný vzor a zároveň se proces držel definovaného Markovova řetězce a tedy konvergoval k náhodné matici.

Vzor	n	#iterací	#workerů	spekulace	čas (s)
	100	100 000	1	-	82,39
	100	100 000	4	ano	24,37
	100	100 000	4	ne	26,68
1 0 0	100	100 000	8	ano	14,74
1 1 1	100	100 000	8	ne	16,70
0 0 1	500	10 000	1	-	430,01
	500	10 000	4	ano	152,11
	500	10 000	4	ne	162,65
	500	10 000	8	ano	92,10
	500	10 000	8	ne	121,26

Vzor	n	#iterací	#workerů	spekulace	čas (s)
	100	100 000	1	-	82,39
	100	100 000	4	ano	24,37
	100	100 000	4	ne	26,68
1 0 0	100	100 000	8	ano	14,74
1 1 1	100	100 000	8	ne	16,70
0 0 1	500	10 000	1	-	430,01
	500	10 000	4	ano	152,11
	500	10 000	4	ne	162,65
	500	10 000	8	ano	92,10
	500	10 000	8	ne	121,26

Vzor	n	#iterací	#workerů	spekulace	čas (s)
	100	100 000	1	-	82,39
	100	100 000	4	ano	24,37
	100	100 000	4	ne	26,68
1 0 0	100	100 000	8	ano	14,74
1 1 1	100	100 000	8	ne	16,70
0 0 1	500	10 000	1	-	430,01
	500	10 000	4	ano	152,11
	500	10 000	4	ne	162,65
	500	10 000	8	ano	92,10
	500	10 000	8	ne	121,26

Děkuji za pozornost.

Pravděpodobnost úspěchu iterace

"Heuristicky, pokud se dívám na vzor, který má lineární extremální funkci, tj. každá vygenerovaná matice tvaru $N \times N$ má nejvýš cN jedniček pro nějakou konstantu c, tak to znamená, že v dlouhodobém průměru s pravděpodobností nejvýš $cN/N^2 = c/N$ program jedničku změní na nulu, a tedy v průměru také s pravděpodobností nejvýš c/N program úspěšně změní nulu na jedničku, protože počet jedniček bude zhruba konstantní. A tedy pravděpodobnost neúspěšné změny by měla být aspoň $1 - 2c/N$. Podíl neúspěšných kroků tedy poroste s N. Já bych si přál, aby ten program šlo použít i v situacích, kdy N jsou řádově stovky, a potom pravděpodobnost úspěšné změny bude v jednotkách procent (pro vzory s lineární extremální funkcí)." Vít Jelínek, 2016