Задачи поиска кратчайших путей с использованием GraphBLAS

Рустам Азимов

Поиск кратчайших путей

- Задача поиска кратчайших путей применима в различных областях
  - Картографические сервисы, сети дорог
  - Компьютерные или телекоммуникационные сети (скорость передачи данных, надежность)

Задачи поиска кратчайших путей

- Пусть дан взвешанный направленный граф $G = (V, E, \omega)$, где $\omega : E \to \mathbb{R} \cup \{\infty\}$ — функция весов
- Вес ∞ означает отсутствие ребра
- Расширим определение функции весов до путей $p = (v_0, \ldots, v_k)$

$$\omega(p) = \sum_{i=0}^{k-1} \omega(v_i, v_{i+1})$$

- Тогда для заданных вершин u и v необходимо найти наименьший вес среди путей между этими двумя вершинами (а иногда и сами пути, имеющие такой вес)

$$\min_{p \in P^{uv}} \omega(p)$$

Существование

- Если между вершинами u и v в графе G существует путь, проходящий через отрицательный цикл, то кратчаяшего пути для этих вершин не существует
- Такие сиутации могут быть обнаружены и можно, например, использовать специальное значение для веса: $-\infty$

Единственность

- В графе без отрицательных циклов вес кратчашего пути между двумя вершинами единственнен
- Хотя кратчайших путей, имееющих такой вес может быть несколько
- Если для кратчайшего пути достижим цикл с нулевой суммой весов, то кратчайших путей бесконечно много
- Если же в графе все циклы имеют положительную сумму весов, то все кратчайшие пути будут простыми

Формулировки

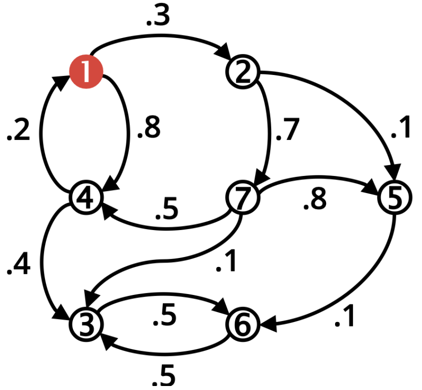
- Мы рассмотрим различные формулировки, отличающиеся количеством стартовых вершин, для которых будут искаться кратчайшие пути
  - SSSP(Single Source Shortest Path) — найти кратчайшие пути из стартовой вершины s до всех остальных вершин
  - MSSP(Multiple-Source Shortest Path) — найти кратчайшие пути из нескольких стартовых вершин до всех остальных
  - APSP(All-Pairs Shortest Path)— найти кратчайшие пути для каждой пары вершин

SSSP: Bellman-Ford

```
Bellman–Ford(V, E, w, s)
foreach v ∈ V
      do d(v) = ∞
         π(v) = NIL
d(s) = 0
for k = 1 to N − 1
      do foreach edge (u, v) ∈ E
            do ▷ Relax (u, v)
               if d(v) > d(u) + W(u, v)
                  then d(v) = d(u) + W(u, v)
                       π(v) = u
foreach edge (u, v) ∈ E
      do if d(v) > d(u) + W(u, v)
            then return "A negative-weight cycle exists."
```

SSSP: Algebraic Bellman-Ford

semiring	domain	⊕	⊗	0
min-plus	$\mathbb{R} \cup \{+\infty\}$	min	+	$+\infty$



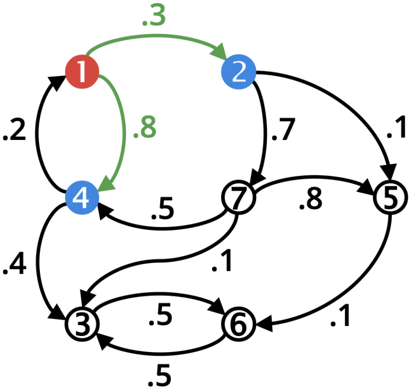
A	①	②	③	④	⑤	⑥	⑦
①	0	.3		.8			
②		0			.1		.7
③			0			.5	
④	.2		.4	0			
⑤					0	.1	
⑥			.5			0	
⑦			.1	.5	.9		0

	❶	②	③	④	⑤	⑥	⑦
d	0						

d min.+ **A**

SSSP: Algebraic Bellman-Ford

semiring	domain	$\oplus$	$\otimes$	0
min-plus	$\mathbb{R} \cup \{+\infty\}$	min	$+$	$+\infty$



A	①	②	③	④	⑤	⑥	⑦
❶	0	.3		.8			
②		0			.1		.7
③			0			.5	
④	.2		.4	0			
⑤					0	.1	
⑥			.5			0	
⑦			.1	.5	.9		0

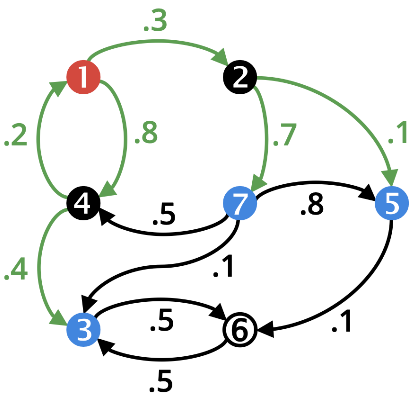
	❶	②	③	④	⑤	⑥	⑦
d	0						

0	.3		.8			

d min.+ **A**

SSSP: Algebraic Bellman-Ford

semiring	domain	⊕	⊗	0
min-plus	$\mathbb{R} \cup \{+\infty\}$	min	$+$	$+\infty$



A	①	②	③	④	⑤	⑥	⑦
❶	0	.3		.8			
❷		0			.1		.7
③			0			.5	
❹	.2		.4	0			
⑤					0	.1	
⑥			.5			0	
⑦			.1	.5	.9		0

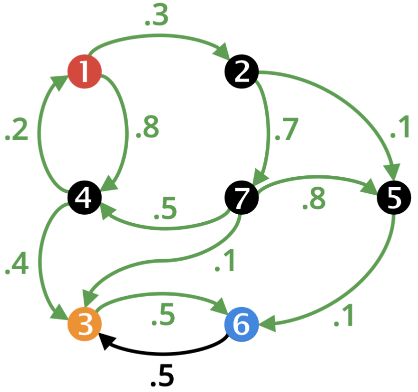
	❶	❷	③	❹	⑤	⑥	⑦
d	0	.3		.8			

0	.3	1.2	.8	.4		1

d min.+ **A**

SSSP: Algebraic Bellman-Ford

semiring	domain	$\oplus$	$\otimes$	0
min-plus	$\mathbb{R} \cup \{+\infty\}$	min	$+$	$+\infty$



A	①	②	③	④	⑤	⑥	⑦
1	0	.3		.8			
2		0			.1		.7
3			0			.5	
4	.2		.4	0			
5					0	.1	
⑥			.5			0	
7			.1	.5	.9		0

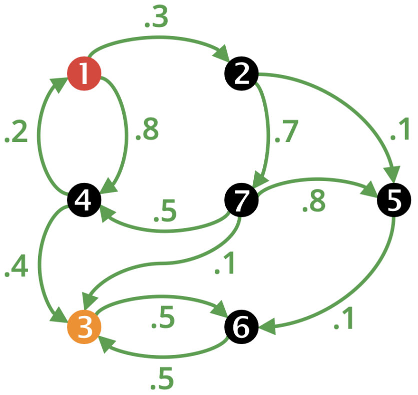
	1	2	3	4	5	⑥	7
d	0	.3	1.2	.8	.4		1

0	.3	1.1	.8	.4	.5	1

d min.+ **A**

SSSP: Algebraic Bellman-Ford

semiring	domain	$\oplus$	$\otimes$	0
min-plus	$\mathbb{R} \cup \{+\infty\}$	min	$+$	$+\infty$



A	①	②	③	④	⑤	⑥	⑦
1	0	.3		.8			
2		0			.1		.7
3			0			.5	
4	.2		.4	0			
5					0	.1	
6			.5			0	
7			.1	.5	.9		0

	1	2	3	4	5	6	7
d	0	.3	1.1	.8	.4	.5	1
d min.+ **A**	0	.3	1	.8	.4	.5	1

SSSP: Algebraic Bellman-Ford

- На итерации k рассматриваем пути длины ровно k
- Поэтому будет подсчитан минимальный вес среди путей длины не более k
- В отличии от BFS тут нету масок, так как повторное посещение вершин может быть полезно
- Однако push/pull direction optimization тут также применима

SSSP: Algebraic Bellman-Ford

```
BELLMAN–FORD(A, s)
d = ∞
d(s) = 0
for k = 1 to N − 1
        do d = d min.+ A
if d ≠ d min.+ A
    then return "A negative-weight cycle exists."
return d
```

APSP: Floyd-Warshall

FLOYD–WARSHALL(V, E, w)

for $u = 1$ **to** N
    do for $v = 1$ **to** N
        do $\mathbf{D}_0(u, v) = \mathbf{W}(u, v)$
            if $u = v$
              then $\Pi_0(u, v) =$ NIL
            if $u \neq v$ and $\mathbf{W}(u, v) < \infty$
              then $\Pi_0(u, v) = u$
              $\triangleright$ else $\Pi_0(u, v)$ is undefined
for $k = 1$ **to** N
    do for $u = 1$ **to** N
        do for $v = 1$ **to** N
            do if $\mathbf{D}_{k-1}(u, k) + \mathbf{D}_{k-1}(k, v) \leq \mathbf{D}_{k-1}(u, v)$
                then $\mathbf{D}_k(u, v) = \mathbf{D}_{k-1}(u, k) + \mathbf{D}_{k-1}(k, v)$
                    $\Pi_k(u, v) = \Pi_{k-1}(k, v)$
                else $\mathbf{D}_k(u, v) = \mathbf{D}_{k-1}(u, v)$
                    $\Pi_k = \Pi_{k-1}(u, v)$

APSP: Floyd-Warshall

- На итерации k рассматриваем пути, в промежуточных (все кроме начальной и конечной) вершинах которых могут быть вершины с номерами $< k$
- С каждой итерацией разрешается использовать все больше вершин в качестве промежуточных, поэтому ограничения на пути ослабевают, пока совсем не исчезнут
- На самой итерации рассматриваются кратчайшие пути в вершину k и из неё, ведь склеив два таких пути мы получим кратчайший путь для нового огранчения

APSP: Algebraic Floyd-Warshal

FLOYD–WARSHALL($\mathbf{A}$)

$\mathbf{D} = \mathbf{A}$
for $k = 1$ **to** N
    do $\mathbf{D} = \mathbf{D}\ .\mathrm{min}\ [\mathbf{D}(:, k)\ \mathrm{min}.{+}\ \mathbf{D}(k, :)]$

APSP: Transitive Closure

- Транзитивное замыкание A^* матрицы смежности A сожержит в себе информацию о всех путях в графе
- В контексте задачи поиска кратчайших путей (APSP) нужно использовать соответсвующее полукольцо, например $min.+$

APSP: Transitive Closure

$$
\begin{aligned}
D &= DA \oplus I \\
&= (DA \oplus I)A \oplus I = DA^2 \oplus A \oplus I \\
&= (DA \oplus I)A^2 \oplus A \oplus I = DA^3 \oplus A^2 \oplus A \oplus I \\
&= \ldots \\
&= DA^{k+1} \oplus A^k \oplus \ldots \oplus A^2 \oplus A \oplus I.
\end{aligned}
$$

- Для чего мы использовали единичную матрицу I?

APSP: Transitive Closure

- Если вычисления сходятся, то

$$A^* = \lim_{k \to \infty} I \oplus A \oplus A^2 \oplus \ldots \oplus A^k$$

- Если операция $\oplus$ идемпотентна ($\forall x \in S, x \oplus x = x$), то

$$(I \oplus A)^k = \bigoplus_{i=1}^{k} A^i$$

- Можем применить технику *repeated squaring* и вычислить A^*, если знаем, что вычисления сходятся за конечное число итераций
- Что насчёт полукольца *min.+*?

Источники

- John S. Baras, George Theodorakopoulos — "Path Problems in Networks"
- Книга: Kepner J., Gilbert J. — "Graph algorithms in the language of linear algebra"
- Презентация: Gábor Szárnyas — "Introduction to GraphBLAS"