Przedmiot: Matematyka w projektowaniu sieci i systemów
Prowadzący: Piotr Chołda piotr.cholda@agh.edu.pl
Kierunek: Teleinformatyka
Semestr: II sem. (zimowy) studiów magisterskich

Konspekt

Piotr Chołda

17 października 2017

1 Algorytmy definiowane na grafach

1.1 Przykładowe problemy optymalizacji kombinatorycznej na grafach

1. Pojęcia programowania sieciowego i optymalizacji kombinatorycznej.

2. Problem poszukiwania minimalnego drzewa rozpinającego (MST, *minimum spanning tree*):

    - algorytm Kruskala:

        procedure KRUSKAL($G = (V, E, f)$)
        ▷ Krawędzie MST: T
            $T \leftarrow \emptyset$
            $\mathcal{E} = E$
            while $|T| < |V| - 1$ **do**
                $e \leftarrow \arg\min_{i \in \mathcal{E}}\{f(i)\}$
                ▷ e: „najlżejsza" krawędź w zbiorze nieuwzględnionych krawędzi
                $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e\}$
                if $T \cup \{e\}$ nie zawiera cyklu **then**
                    $T \leftarrow T \cup \{e\}$
                end if
            end while
        end procedure

    - algorytm Prima (zwany również algorytmem Prima-Dijkstry):

        procedure PRIM($G = (V, E, f)$, $r \in V$)
        ▷ Krawędzie MST: T
        ▷ r: korzeń
            $T \leftarrow \emptyset$
            $\mathcal{V} = V \setminus \{r\}$
            while $\mathcal{V} \neq \emptyset$ **do**
                $\mathcal{L} = \{i \in E : i = \{t, v\}, t \in V \setminus \mathcal{V}, v \in \mathcal{V}\}$
                $e \leftarrow \arg\min_{i \in \mathcal{L}}\{f(i)\}$
                ▷ e: „najlżejsza" krawędź w zbiorze krawędzi, które powiększą drzewo
                $\mathcal{V} \leftarrow \mathcal{V} \setminus \{v \in \mathcal{V} : e = \{t, v\}\}$
                $T \leftarrow T \cup \{e\}$

end while
end procedure

3. Pojęcie algorytmu zachłannego (*greedy algorithm*).

4. Przykłady „trudnych” problemów optymalizacyjnych:
   - problem Steinera,
   - problem kolorowania (barwienia) wierzchołków w grafie; twierdzenie o czterech barwach, pojęcie grafu planarnego, genusa, wzoru Eulera.

5. Algorytm przeszukiwania grafu wszerz (BFS, *Breadth-First Search*), drzewo ścieżek najkrótszych pod względem liczby przeskoków (*hops*):

procedure BFS($G = (V, A)$, $r \in V$)
▷ r: korzeń
▷ Inicjalizacja:
$\mathcal{S} \leftarrow \{r\}$
▷ $\mathcal{S}$: zbiór wierzchołków, do których istnieje ścieżka skierowana od korzenia
$\mathcal{L} \leftarrow (r)$
▷ $\mathcal{L}$: uporządkowana lista przeszukiwanych wierzchołków
$\mathcal{L}' \leftarrow V \setminus \{r\}$
▷ $\mathcal{L}'$: zbiór dotychczas nieprzeszukanych wierzchołków
$predecessor(r) = 0$
▷ $predecessor(j) = k$ oznacza, że poprzednikiem wierzchołka j na ścieżce skierowanej od korzenia r jest wierzchołek k
▷ Korzeń nie ma poprzednika
▷ Pętla główna:
while $\mathcal{L} \neq \emptyset$ **do**
for all $k \in \mathcal{L}$ **do**
for all $j \in \mathcal{L}'$ **do**
if $(k, j) \in A$ **then**
$\mathcal{S} \leftarrow \mathcal{S} \cup \{j\}$
$predecessor(j) = k$
$\mathcal{L} \leftarrow (\mathcal{L}, j)$
$\mathcal{L}' \leftarrow \mathcal{L}' \setminus \{j\}$
end if
end for
$\mathcal{L} \leftarrow \mathcal{L} \setminus \{k\}$
end for
end while
return ($\mathcal{S}$,$\mathcal{P}(\mathcal{S})$)
▷ $\mathcal{P}(\mathcal{S})$: lista poprzedników wierzchołków należących do zbioru $\mathcal{S}$
end procedure

6. Algorytm przeszukiwania grafu w głąb (DFS, *Depth-First Search*):

procedure DFS($G = (V, A)$, $r \in V$)
▷ r: korzeń
▷ Inicjalizacja:
$\mathcal{S} \leftarrow \{r\}$

    ▷ $\mathcal{S}$: zbiór wierzchołków, do których istnieje ścieżka skierowana od korzenia
    $\mathcal{L}' \leftarrow V \setminus \{r\}$
    ▷ $\mathcal{L}'$: zbiór dotychczas nieprzeszukanych wierzchołków
    $\mathit{predecessor}(r) = 0$
    ▷ Pętla główna:
    SearchDeep(r,G,$\mathcal{S}$,$\mathcal{L}'$)
    return ($\mathcal{S}$,$\mathcal{P}(\mathcal{S})$)
end procedure

Procedura wykonywana rekurencyjnie:

procedure SearchDeep(v,G,$\mathcal{S}$,$\mathcal{L}'$)
    for all $j \in \mathcal{L}'$ **do**
        if $(v, j) \in A$ **then**
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{j\}$
            $\mathit{predecessor}(j) = v$
            $\mathcal{L}' \leftarrow \mathcal{L}' \setminus \{j\}$
            SearchDeep(j,G,$\mathcal{S}$,$\mathcal{L}'$)
        end if
    end for
end procedure

7. Problem poszukiwania przepływu maksymalnego. Twierdzenie Forda-Fulkersona.

1.2 Zadania

- Proszę podać przykład takiego grafu ważonego G, który spełnia na raz wszystkie poniższe warunki:
  - ⋆ graf G jest spójny,
  - ⋆ wszystkie wagi są liczbami naturalnymi,
  - ⋆ graf G ma dziewięć wierzchołków,
  - ⋆ suma wag minimalnego drzewa rozpinającego grafu G jest równa połowie sumy wszystkich wag w tym grafie.
- Proszę pokazać, że minimalne drzewo rozpinające grafu pełnego K_9 o dowolnych wagach jest grafem dwudzielnym.

1.3 Lektury

1.3.1 Materiał wykładu

Zagadnienia omówione w ramach tego wykładu są w dużym stopniu opisane w następujących książkach:

- Maciej M. Sysło, Narsingh Deo, and Janusz S. Kowalik. *Algorytmy optymalizacji dyskretnej*. Wydawnictwo Naukowe PWN, Warszawa, 1999: rozdziały 3.4-3.6.
- Robin J. Wilson. *Wprowadzenie do teorii grafów*. Wydawnictwo Naukowe PWN, Warszawa, 2000: § 17, § 29.

1.3.2 Bibliografia uzupełniająca

- Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. Minimum Cost Flow Problem. In Christodoulos A. Floudas and Panos M. Pardalos, editors, *Encyclopedia of Optimization*, pages 2095–2108. Springer Science+Business Media, LLC., New York, NY, 2009: poszukiwanie przepływów o najniższym koszcie.

- Maciej M. Sysło, Narsingh Deo, and Janusz S. Kowalik. *Algorytmy optymalizacji dyskretnej*. Wydawnictwo Naukowe PWN, Warszawa, 1999: podstawy teoretyczne do naszego kursu.

- Robin J. Wilson. *Wprowadzenie do teorii grafów*. Wydawnictwo Naukowe PWN, Warszawa, 2000: zwięzłe wprowadzenie do teorii grafów, trochę algorytmów.