Sposób rozwiązywania przykładowego zadania ze sprawdzianu z użycia OPL/CPLEX

Kroki postępowania

1. Podpisanie się na kartce.
2. Analiza zadania polegająca na wyodrębnieniu treści ważnych z punktu widzenia modelowania optymalizacyjnego.
3. Zapisanie modelu matematycznego i naniesienie go na kartkę z zadaniem.
4. Zapisanie pliku z modelem.
5. Zapisanie pliku z danymi.
6. Wysłanie e-maila z rozwiązaniem do prowadzących oraz oddanie kartki.

Analiza zadania

Najpierw należy zwrócić uwagę na istotne informacje, które posłużą do wypracowania modelu matematycznego zadania optymalizacji — tj. na elementy, które należy wyabstrahować z „historyjki":

Na potrzeby rozgłaszania informacji **do wszystkich** użytkowników (**węzłów**) **sieci** administrator tworzy tunel (połączenie wirtualne), w którym **dane rozchodzą się po kolei od jednego węzła do drugiego**, w taki sposób że rozgłaszana **informacja nie wraca już do węzła, któremu została dostarczona**, ponadto **na koniec informacja wraca do węzła źródłowego** (w celu sprawdzenia, czy istotnie dostarczono ją do wszystkich innych węzłów). W celach niezawodnościowych administrator utworzy **dwa tunele**, które **nie będą się w ogóle nakładać** — chodzi o to, żeby zapewnić poprawną transmisję co najmniej jednym tunelem w przypadku, gdy **uszkodzeniu ulegnie któreś z łączy sieci**. **Znane są również opóźnienia łączy**. Administrator chciałby, żeby **opóźnienia w dostarczaniu informacji były jak najmniejsze**. Jak utworzyć te tunele?

Interpretacja wyróżnionych fragmentów:

- *do wszystkich ... węzłów sieci ... dane rozchodzą się po kolei od jednego węzła do drugiego ... informacja nie wraca już do węzła, któremu została dostarczona ... na koniec informacja wraca do węzła źródłowego*: tunel to po prostu cykl Hamiltona (cykl: zamknięta ścieżka, w której nie powtarzają się krawędzie, cykl Hamiltona: obejmuje wszystkie wierzchołki);
- *dwa tunele ... nie będą się w ogóle nakładać ... uszkodzeniu ulegnie któreś z łączy sieci*: chodzi o dwa cykle Hamiltona, które są rozłączne z punktu widzenia krawędzi sieci;
- *znane są również opóźnienia łączy*: podano pewną wagę dla krawędzi;
- *opóźnienia w dostarczaniu informacji były jak najmniejsze*: chodzi o minimalizację tej wagi z punktu widzenia znalezionych cykli — tutaj trzeba wiedzieć, że opóźnienie sumuje się na poszczególnych łączach, czyli jest miarą addytywną.

Zapis matematyczny problemu

- Wiadomo, że należy użyć programowania matematycznego LP, ILP lub MILP. Państwo znają sformułowanie *problemu komiwojażera*, który służy do znalezienia najkrótszego cyklu Hamiltona przy wadze addytywnej — jest to sformułowanie MILP (używamy pomocniczej zmiennej ciągłej d^x, żeby sobie wyliczyć opóźnienie, co później przyda się przy wyświetlaniu wyniku w eleganckiej postaci):

$$\min d^x$$
$$\text{s.t.:} \begin{cases} \sum\limits_{j:(i,j)\in A} x_{(i,j)} = 1 \quad i \in V \\ \sum\limits_{j:(j,i)\in A} x_{(j,i)} = 1 \quad i \in V \\ \sum\limits_{\substack{i,j\in Z \\ (i,j)\in A}} x_{(i,j)} \leq |Z| - 1 \quad Z \subset V, Z \neq \emptyset \\ d^x = \sum\limits_{(i,j)\in A} d_{(i,j)} x_{(i,j)} \\ x_{(i,j)} \in \mathbb{B} \\ d^x \in \mathbb{R}_+ \end{cases}$$

- Znane sformułowanie należy poszerzyć o następujące elementy:
  - ⋆ jednocześnie używamy drugiego cyklu:
    - ∗ dodamy analogiczne ograniczenia dla drugiego cyklu:
$$\begin{cases} \sum\limits_{j:(i,j)\in A} y_{(i,j)} = 1 \quad i \in V \\ \sum\limits_{j:(j,i)\in A} y_{(j,i)} = 1 \quad i \in V \\ \sum\limits_{\substack{i,j\in Z \\ (i,j)\in A}} y_{(i,j)} \leq |Z| - 1 \quad Z \subset V, Z \neq \emptyset \\ d^y = \sum\limits_{(i,j)\in A} d_{(i,j)} y_{(i,j)} \\ y_{(i,j)} \in \mathbb{B} \\ d^y \in \mathbb{R}_+ \end{cases}$$
    - ∗ do funkcji celu należy jeszcze dodać wagę drugiego cyklu: d^y;
  - ⋆ dbamy o rozłączność obu cykli, co oznacza że na żadnej z krawędzi nie mogą znajdować się jednocześnie oba cykle (ale np. mogą być krawędzie, gdzie w ogóle nie przebiega żaden z cykli), co oznacza, że dla każdej krawędzi zakazujemy, aby obie zmienne oznaczające używanie tej krawędzi przez cykl nie były na raz jedynkami — dodajemy ograniczenie:
    $x_{(i,j)} + y_{(i,j)} \leq 1 \quad (i,j) \in A$
    .

- W związku z tym całość sformułowania (należy je zapisać na drugiej stronie kartki z zadaniem):
$$\min d^x + d^y$$
$$\text{s.t.:} \begin{cases} \sum\limits_{j:(i,j)\in A} x_{(i,j)} = 1 \quad i \in V \\ \sum\limits_{j:(j,i)\in A} x_{(j,i)} = 1 \quad i \in V \\ \sum\limits_{\substack{i,j\in Z \\ (i,j)\in A}} x_{(i,j)} \leq |Z| - 1 \quad Z \subset V, Z \neq \emptyset \\ d^x = \sum\limits_{(i,j)\in A} d_{(i,j)} x_{(i,j)} \\ \sum\limits_{j:(i,j)\in A} y_{(i,j)} = 1 \quad i \in V \\ \sum\limits_{j:(j,i)\in A} y_{(j,i)} = 1 \quad i \in V \\ \sum\limits_{\substack{i,j\in Z \\ (i,j)\in A}} y_{(i,j)} \leq |Z| - 1 \quad Z \subset V, Z \neq \emptyset \\ d^y = \sum\limits_{(i,j)\in A} d_{(i,j)} y_{(i,j)} \\ x_{(i,j)} + y_{(i,j)} \leq 1 \quad (i,j) \in A \\ x_{(i,j)}, y_{(i,j)} \in \mathbb{B} \\ d^x, d^y \in \mathbb{R}_+ \end{cases}$$

Implementacja w OPL/CPLEX

Należy utworzyć sobie nowy projekt CPLEX, a potem zacząć implementację od pliku `.mod`. Wynika to z faktu, że opracowanie modelu jest zazwyczaj trudniejsze koncepcyjnie niż wprowadzenie danych i będzie bardziej doceniane. Plik z modelem wygląda następująco:

```
{string} Nodes = ...; // wezly/wierzcholki sieci

tuple arc
{
  string source;
  string destination;
```

```
}

{arc} Arcs with source in Nodes, destination in Nodes = ...; // lacza/krawedzie
    (wlasciwie: luki) sieci

float d[Arcs] = ...; // opoznienia na laczach

int number_nodes = card(Nodes);
range number_subsets_nodes = 1..(ftoi(pow(2,number_nodes))-2);
{string} power_sets_Nodes[k in number_subsets_nodes] = {i | i in Nodes: ((k div
    (ftoi(pow(2,ord(Nodes,i)))) mod 2) == 1)};
{arc} arcs_power_setNodes[k in number_subsets_nodes] = {<i,j> | <i,j> in Arcs: i
     in power_sets_Nodes[k] && j in power_sets_Nodes[k]};

{string} neighbIn[i in Nodes] = {j | <i,j> in Arcs};
{string} neighbOut[i in Nodes] = {j | <j,i> in Arcs};

dvar boolean x[Arcs]; // zmienna wskazujaca, ktore lacza tworza pierwszy tunel
dvar boolean y[Arcs]; // zmienna wskazujaca, ktore lacza tworza drugi tunel

dvar float+ delx; //sumaryczne opoznienie w pierwszym tunelu
dvar float+ dely; //sumaryczne opoznienie w drugim tunelu

minimize delx + dely;

subject to{

  forall(i in Nodes)
   sum(j in neighbIn[i]) x[<i,j>] == 1;

  forall(i in Nodes)
   sum(j in neighbOut[i]) x[<j,i>] == 1;

  forall(k in number_subsets_nodes)
     sum(a in arcs_power_setNodes[k]) x[a] <= card(power_sets_Nodes[k])-1;

  delx == sum(a in Arcs) d[a]*x[a];

  forall(i in Nodes)
   sum(j in neighbIn[i]) y[<i,j>] == 1;

  forall(i in Nodes)
   sum(j in neighbOut[i]) y[<j,i>] == 1;

  forall(k in number_subsets_nodes)
     sum(a in arcs_power_setNodes[k]) y[a] <= card(power_sets_Nodes[k])-1;

  dely == sum(a in Arcs) d[a]*y[a];

  forall(a in Arcs)
     x[a] + y[a] <= 1;
}
```

W związku z tym, że oczekuje się eleganckiego przedstawienia wyników rozwiązania, należy dodać krótki skrypt, który inteligentnie poda wynik — np. wyświetli łącza używane przez poszczególne tunele oraz poda opóźnienia na nich:

```
execute{

 writeln("Lacza tworzace pierwszy tunel o sumarycznym opoznieniu ",delx,":");

 for(var a in Arcs)
```

```
    if(x[a] > 0) writeln("lacze: ",a.source,"-",a.destination);

  writeln("Lacza tworzace drugi tunel o sumarycznym opoznieniu ",dely,":");

  for(var a in Arcs)
    if(y[a] > 0) writeln("lacze: ",a.source,"-",a.destination);

}
```

Potem można wprowadzić dane w pliku `.dat`. Warto pamiętać o tym, że dla konkretnego studenta sieć jest wylosowana, podobnie jak wartości wag. W związku z czym pojawienie się sprzeczności w rozwiązaniu wcale nie oznacza, że popełniło się błąd w modelu. Plik danych może wyglądać następująco:

```
Nodes = {A, B, C, D, E, F};

Arcs = {<A,B>, <B,A>, <B,C>, <C,B>, <A,F>, <F,A>, <A,E>, <E,A>, <B,E>, <E,B>, <E
    ,C>, <C,E>, <C,D>, <D,C>, <F,E>, <E,F>, <E,D>, <D,E>, <D,F>};

d = [1 14 3 15 10 8 17 19 12 11 2 14 3 15 4 18 3 19 2];
```

Oddawanie rozwiązania

Na koniec należy:

- wysłać paczkę z projektem do prowadzących (projekt ma być spakowany, a e-mail powinien mieć nadany odpowiedni tytuł);
- oddać kartę z zadaniem.