Aalto University

Lecture Notes - Week V

NP Problems and Graphs - Part I

Fernando Dias, Philine Schiewe and Piyalee Pattanaik

January 8, 2024

CHAPTER 1
Decision Problems

So far, all problems either have **algorithm** or **a mixed integer** formulation.

However, what happens when **an algorithm** is not achievable nor efficient? A mixed integer formulation **might still** be possible, but would it be enough?

In simple terms, a decision problem is a "yes-or-no question" on an infinite set of possible solutions. Different types of inputs can be used, such as natural numbers, binary strings or any string. For those inputs in which the decision is "yes", the subset is called a formal language.

The most common case of a decision problem is a set of prime numbers. It is straightforward to test if a given number is prime.

If $\mathcal{K}$ is the problem, and x is the input, we will often write $x \in \mathcal{L}$ to denote a yes answer and $y \notin \mathcal{L}$ to denote a no answer. This notation comes from thinking of $\mathcal{L}$ as a **language** and asking whether x is in the language $\mathcal{L}$ (yes) or not (no).

In a **decision problem:** a pair (X, Y) where X is a language decidable in polynomial time and $Y \subset X$:

- *instance:* $x \in X$
- *yes-instance:* $x \in Y$
- *no-instance:* $x \in X \setminus Y$
- an algorithm for a decision problem computes function $f\colon X \to \{0,1\}$ with $f(x) = 1 \iff x \in X$

They differ from **optimization problem** because the former requires an answer that has an **optimal configuration**.

For instance: "*What is the shortest path between two nodes?*" **vs** *"Is a particular path P the shortest path between these two nodes?*".

Remark: An optimization problem has a **corresponding** decision problem.

1.1 CLASS P

Amongst different problems, the **class** P is the **class** of all decision problems (X, Y) for which there is a **polynomial** algorithm (in terms of runtime). Hence, given $x \in A^*$, compute $f(x) \in \{0, 1\}$ with $\text{time}(x) \leq p(\text{size}(x))$.

Examples:

- linear inequalities;
- shortest path;
- maximum matching;
- minimum cost flow.

In this course, **all problems we have discussed so far fall into this category**.

1.2 CLASS NP

A decision problem (X, Y) belongs to class NP if for each $y \in Y$ a certificate c can be verified in **polynomial time**. Those certificates are usually a **feasible solution** to the problem.

The name $NP =$ **nondeterministic** polynomial: "guessing" certificates long enough would work, and (X, Y) can be solved by **nondeterministic in polynomial time**.

A few examples are:

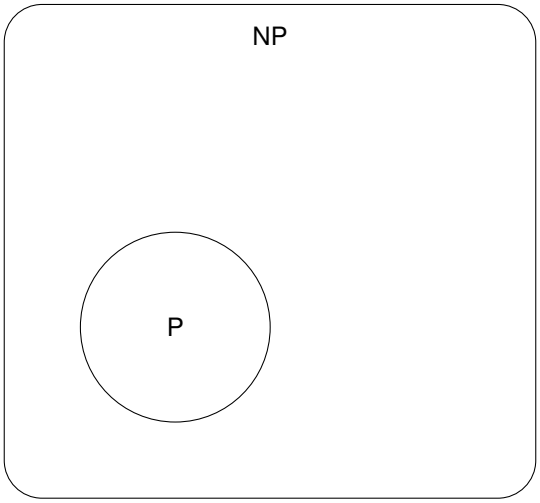
- integer linear inequalities $\rightarrow$ c is feasible vector x
- knapsack $\rightarrow$ c is a feasible set of items to take

One of the biggest question in Discrete Mathematics: Is $P = NP$?

Theorem 1 *$P \subseteq NP$.*

Proof 1 *$(X, Y) \in P$ can be decided in polynomial time. $\Rightarrow$ x can be used as **certificate**.*

The most acceptable answer is given in the following polynomial:



So far, we have studied many problems in the class P extensively. Now, we are pivoting towards some problems in the class NP.

CHAPTER 2

Travelling Salesman Problem (TSP)

2.1 DEFINITION AND HISTORY

First, setting up the definition of this problem: imagine a scenario where a set of cities are expecting a visit from a **travelling** merchant.

As part of their visit, this **salesman** has to start and finish their travel in the same city, cannot visit the same city more than once, and every city has to be visited in a single trip.

Figure 2.1: A lone traveller about to make important decisions

→ What is the shortest possible route?

Such a problem is called Travelling Salesman Problem, very important to the fields of theoretical computer science and operations research.

It was first described by Irish mathematician **W.R. Hamilton** and British mathematician **Thomas Kirkman** in the 1800s through the description of a game where the solution involved a cycle without overlapping nodes.


W. R. Hamilton


Thomas Kirkman

2.2 INITIAL APPROACH

At first glance, the first solution is to try all possibilities and choose the best solution. This process is called **enumeration**, and it is extremely time-consuming. Furthermore, with larger instances, more time is required.

Initial challenge: for an instance with n cities, there are 2^n possible combinations. Hence impractical.

Is there any **better alternative**?

First, let us assume that the set of cities can be modelled as **graph** $G = (V, E, f)$, where:

- V is the set of **individual cities**;
- E represent the **paths** between a **pair** of cities and;
- f_{ij} is the **cost** to travel from city i to city j, for all $(i, j) \in E$.

The choice to travel from city i to city j using a path (edge in our modelling) connecting them is modelled by our decision variable.

$$x_{ij} = \left\{ \begin{array}{ll} 1, & \text{if path goes from city } i \text{ to city } j \\ 0, & \text{otherwise} \end{array} \right\}$$

Our objective function can also be derived from the problem description:

$$\min \sum_{i}^{n} \sum_{j,j \neq i}^{n} x_{ij} f_{ij}$$

where n is the number of cities ($|V| = n$).

Regarding limitations, two constraints can also be derived directly from the description: *"cannot visit the same city more than once"*. Hence, each city requires **one in-edge** and **one out-edge**.

Hence, the following constraints:

- **Singular** incoming degree:

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 \quad \forall j \in \{1, \cdots, n\}$$

- **Singular** outgoing degree:

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1 \quad \forall i \in \{1, \cdots, n\}$$

Those constraints characterize **paths** in our solution (similar to the constraints in the shortest path (ILP formulation)).

2.3 SUBTOUR ELIMINATION

As mentioned, these constraints imposed that every city is visited only once. However, they do not guarantee that **there is a single trip that will connect all cities**.

For instance:

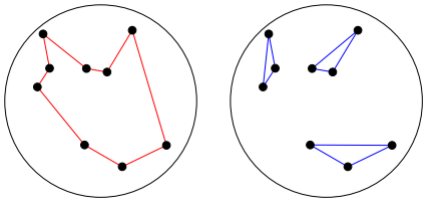


Figure 2.2: Two solutions that do not violate the previous constraints, but only one has a single trip.

This is the main reason why TSP is a challenging problem.

There are two main strategies to prevent separate tours (subtour) from our potential solution: Miller-Tucker-Zemlin and Dantzig-Fulkerson-Johnson. Both impose the presence of a single tour using **linear constraints**.

2.3.1 Miller-Tucker-Zemlin

The formulation described by Miller-Tucker-Zemlin in 1960 requires an additional variable to track which city has been visited starting from initial city $i = 1$. By setting $u_j > u_i$, it determines the **order** of visiting each city (city j will be visited after city i).

This leads to the following requirement:

$$u_j \geq u_i + 1 \text{ if } x_{ij} = 1$$

which can encapsulated as the following constraints:

$$u_i - u_j + 1 \leq (n-1)(1-x_{ij}) \qquad \text{for } 2 \leq i \neq j \leq n$$
$$2 \leq u_i \leq n \qquad \text{for } 2 \leq i \leq n$$

The resulting formulation is given below:

$$\text{Minimize} \quad \sum_{i}^{n} \sum_{j,j\neq i}^{n} x_{ij} f_{ij}$$

Subject to:

$$\sum_{i=1,i\neq j}^{n} x_{ij} = 1 \qquad \forall j \in \{1, \cdots, n\}$$
$$\sum_{j=1,j\neq i}^{n} x_{ij} = 1 \qquad \forall i \in \{1, \cdots, n\}$$
$$u_i - u_j + 1 \leq (n-1)(1-x_{ij}) \qquad 2 \leq i \neq j \leq n$$
$$2 \leq u_i \leq n \qquad 2 \leq i \leq n$$
$$x_{ij} \in \{0,1\} \qquad i,j \in \{1, \cdots, n\}$$

2.3.2 Dantzig-Fulkerson-Johnson

An alternative (and earlier formulation from Dantzig, Fulkerson and Johnson in 1954) imposes an extra requirement that eliminates all subset of nodes to create a subtour.

$$\sum_{i \in Q} \sum_{j \neq i, j \in Q} x_{ij} \leq |Q| - 1 \qquad \forall Q \subset \{1, \cdots, n\}, |Q| \geq 2$$

By using the equation above, the resulting model is:

$$\text{Minimize} \quad \sum_{i}^{n} \sum_{j,j\neq i}^{n} x_{ij} f_{ij}$$

Subject to:

$$\sum_{i=1,i\neq j}^{n} x_{ij} = 1 \qquad \forall j \in \{1, \cdots, n\}$$
$$\sum_{j=1,j\neq i}^{n} x_{ij} = 1 \qquad \forall i \in \{1, \cdots, n\}$$
$$\sum_{i \in Q} \sum_{j \neq i, j \in Q} x_{ij} \leq |Q| - 1 \qquad \forall Q \subset \{1, \cdots, n\}, |Q| \geq 2$$
$$x_{ij} \in \{0,1\} \qquad i,j \in \{1, \cdots, n\}$$

This formulation is numerically and empirically stronger, but both are useful regardless.