# 1 Workflow Satisfiability Problem

The workflow Satisfiability Problem is an important problem in access control (information security). It is about organising a workflow in an organisation in such a way that certain security-related constraints are satisfied. You are given a set $U$ of employees, or users, and a set $S$ of tasks, or steps (e.g., a task could be to check a contract, or to sign it, or to send it to the client). The problem is to find an assignment $\pi : S \to U$ of users to steps such that all the constraints are satisfied. (Note that every step has to be assigned exactly one user; a user may be assigned one or several steps, or may not be assigned any steps.)

The constraints can be of the following types:

- Authorisations: a user can only be assigned steps that they are authorised for; for example, an employee may not be qualified to check a contract, and hence will not be authorised for the corresponding step. If authorisations are not specified for a user, that user can be assigned any steps. Authorisations can be specified at most once for each user.

  **Formal definition:**
  Given: a user $u \in U$ and an authorisation list $A \subseteq S$. Requirement: if $\pi(s) = u$ for some $s \in S$ then $s \in A$.

- Separation of duty: for a given pair of tasks, ensure that they are assigned to two different employees; for example, if a contract requires two signatures, those signatures have to come from two different people

  **Formal definition:**
  Given: a pair of steps $s' \neq s'' \in S$.

  Requirement: $\pi(s^{'}) \neq \pi(s^{''})$.

- Binding of duty: for a given pair of tasks, ensure that they are assigned the same employee; for example, the person responsible for checking a contract should also sign that contract.

  **Formal definition:**
  Given: a pair of steps $s' \neq s'' \in S$.

  Requirement: $\pi(s^{'}) \neq \pi(s^{''})$.

  *When you read this document for the first time, I suggest that you skip the rest of the constraints.*

- At-most-$k$: for a given set of steps $T \subseteq S$, ensure that at most $k$ users are assigned to steps $T$. For example, when several tasks are associated with a piece of sensitive information, it is good to keep the number of people having access to it to a minimum.

  **Formal definition:**
  Given: a set of steps $T \subseteq S$ and a positive integer $k \leq |S|$.

  Requirement: $|\{\pi(s) : s \in T\}| \leq k$.

- One-team constraint: for a given set of steps and a list of teams, ensure that all these steps are assigned to members of one team.

  **Formal definition:**
  Given: a set of steps $T \subseteq S$; given a set of teams $U_1, U_2, ..., U_r$, where $U_r \subseteq U$ for $i = 1,2,...,r$ and $U_i \cap U_j = \emptyset$ for every $i \neq j \in \{1,2,...,r\}$.

Requirement: $\{\pi(s) : s \in T\} \subseteq U_i$ for some $i \in \{1,2,...,r\}$.

Example of a problem instance with four users $U = \{u_1,u_2,u_3,u_4\}$, three steps $S = \{s_1, s_2, s_3\}$ and six constraints:

1. Authorisations: $u = u_1$ and $A = \{s_1, s_2\}$.
2. Authorisations: $u = u_2$ and $A = \{s_3\}$.
3. Authorisations: $u = u_4$ and $A = \{s_3\}$.
4. Binding of duty: $s' = s_1$, $s''= s_3$.
5. Separation of duty: $s' = s_1$, $s'' = s_2$.
6. Separation of duty: $s' = s_2$, $s'' = s_3$.

Solution: $\pi(s_1) = u_3$, $\pi(s_2) = u_1$, $\pi(s_3) = u_3$.

In fact, this is the only solution to this problem instance.

## 2 Input file format

The input data will be provided in text files. The file begins with a header of the following format:

#Steps: 3
#Users: 4
#Constraints: 6

Steps are referred to as $s$X, where X is the index of the step. In the above example, the problem instance has three steps: $s1$, $s2$, and $s3$. Indexing begin with 1.

Similarly, users are referred to as $u$X, where X is the index of the user. In the above example, the problem instance has four users: $u1$, $u2$, $u3$ and $u4$.

The header is followed by $c$ lines, where $c$ is the number of constraints given in the header (six in this example). Each line defines exactly one constraint.

Each constraint line begins with the constraint name, followed by parameters. The parameters of each constraint are explained below.

- Example of an authorisations constraint:

  Authorisations $u1$ $s1$ $s2$

  This defines an authorisations constraint for $u = u_1$ and $A = \{\ s_1, s_2\}$

The list of steps can be empty, e.g.

Authorisations $u1$

This means that $u_1$ is not authorised for any steps.

- Example of a binding-of-duty constraint:

  Binding-of-duty $s1$ $s3$

  This defines a binding-of-duty constraint for $s'$ = $s_1$, $s''$ = $s_3$.

- Example of a separation-of-duty constraint:

  Separation-of-duty $s1$ $s2$

  This defines a separation-of-duty constraint for $s'$ = $s_1$, $s''$ = $s_2$.

- Example of an at-most-*k* constraint:

  At-most-k 2 $s1$ $s2$ $s3$

  This defines an at-most-*k* constraint for *k* = 2 and *T* = $\{s_1, s_2, s_3\}$.

- Example of a one-team constraint:

  One-team $s1$ $s2$ $s4$ ($u1$ $u2$) ($u5$ $u7$ $u8$) ($u3$)

  This defines a one-team constraint for *T* = $\{s_1, s_2, s_4\}$ and *r* = 3 teams $U_1$ = $\{u_1, u_2\}$, $U_2$ = $\{u_5, u_7, u_8\}$ and $U_3$ = $\{u_3\}$.

  Each team has to include at least one user, and the set of steps has to include at least one step. The number of teams can be anything from 1 to |*U*|.

The format is not case-sensitive, e.g. it is legal to write Authorisations or authorisations. Extra spaces between words, numbers etc. should be ignored when reading the file, e.g. it is legal to define a constraint as follows:

```
Authorisations          u1       s1 s2.
```

You can assume that the input file is correctly formatted.

The following file describes the instance given in Section 2.

```
#Steps: 3
#Users: 4
#Constraints: 6
Authorisations u1 s1 s2
Authorisations u2 s3
Authorisations u4 s3
```

Appendix 1: Workflow Satisfiability Problem

Binding-of-duty $s1$ $s3$
Separation-of-duty $s1$ $s2$
Separation-of-duty $s2$ $s3$

You can find this instance in file example3.txt.

The samples instances can be downloaded in Moodle

List of the main sample instances:

| # | Auth. | BOD | SOD | At-most-$k$ | One-team | Sat | Unique |
|---|---|---|---|---|---|---|---|
| Example1 | ✓ | | | | | ✓ | |
| Example2 | ✓ | | | | | | |
| Example3 | ✓ | ✓ | ✓ | | | ✓ | ✓ |
| Example4 | ✓ | ✓ | ✓ | | | | |
| Example5 | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ |
| Example6 | ✓ | ✓ | ✓ | ✓ | | | |
| Example7 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Example8 | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| Example9 | ✓ | ✓ | ✓ | ✓ | | ✓ | |
| Example10 | ✓ | ✓ | ✓ | ✓ | | ✓ | |
| Example11 | ✓ | ✓ | ✓ | ✓ | | ✓ | |
| Example12 | ✓ | ✓ | ✓ | ✓ | | ✓ | |
| Example13 | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| Example14 | ✓ | ✓ | ✓ | | | | |
| Example15 | ✓ | ✓ | ✓ | | | | |
| Example16 | ✓ | ✓ | ✓ | ✓ | | ✓ | |
| Example17 | ✓ | ✓ | ✓ | ✓ | | ✓ | |
| Example18 | ✓ | ✓ | ✓ | ✓ | | | |
| Example19 | ✓ | ✓ | ✓ | ✓ | | | |

The columns of the table are as follows (from left to right):

- Instance name
- Whether it includes authorisation constraints
- Whether it includes binding-of-duty constraints
- Whether it includes separation-of-duty constraints

- Whether it includes at-most-*k* constraints
- Whether it includes one-team constraints
- Whether it is satisfiable
- Whether there exists exactly one solution

Examples 1–8 are small and good for debugging. Examples 9–15 are relatively large and good for testing. Examples 16–19 are large and good for performance analysis.

In addition to these instances, you can use the instances in folders ‘3-constraint’, ‘4-constraint’ and ‘5-constraint’. Instances in each of these folders share similar properties and thus are good for empirical analysis of your solver.