

# Robust Operations Research III: Operations

Yvo Desmedt

BT Professor of Information Security

University College London  
UK

March 10, 2006

# OVERVIEW

1. Robustness
2. Prior work
3. Two examples
4. A general approach
5. An illustration
6. Variants

# 1. ROBUSTNESS

Examples:

- Robust Threshold Cryptography: Cryptography that continues to work when insiders behave dishonestly
- Robust Control: a part of control theory, deals with control systems that are tolerant to changes in the environment, in the system parameters and in the system.

Robustness: guarantee that desired properties remain even if insiders behave dishonestly.

## 2. PRIOR WORK

Most work originates from communication networks, in particular, we distinguish:

- $k$  nodes are destroyed
- $k$  nodes are actively malicious (Byzantine)
- also require (or not): privacy

Extending the reliability (under attack) of computer networks to include the mechanical (production) world:

In 1998 Burmester-Desmedt-Wang proposed a model based on AND/OR graphs to deal with this extension:

- The AND nodes: PERT type role.



- The OR nodes: redundancy, network (water, fuel)

Desmedt-Wang (COCOON 2002) add **flows** to AND/OR graph model and study critical flows:

- The OR nodes: total output-flow=total input flow
- The AND nodes: e.g. 1 car needs multiple wheels, motor, etc.

**Critical capacity:** need enough water, food, fuel, etc.

When  $k$  nodes **are destroyed** does the capacity drop below the critical one?

Desmedt-Burmester-Wang provide a **macro study using an economics model.**



### 3. TWO EXAMPLES

Above is limited to graphs and primarily flows. However operations in our world are not necessarily graph based. We propose robust operations. We give two examples:

#### **Robust knapsacks:**

**Knapsack:** the problem of choosing some of  $n$  items, each with its value and size, to be loaded in one container, such that the total value of the selected items is above a threshold, without exceeding the size of the container.

**Multiple knapsack:** (also called Multiple Loading Problem) is the problem of choosing some of  $n$  items to be loaded in  $m$  distinct containers, such that the total value of the selected items is maximized, without exceeding the size of each of the containers.



**Simple robust multiple knapsack:** The adversary can choose a set  $B$  of up to  $k$  containers to destroy. One must guarantee that the transported value remaining after the attack (regardless which set  $B$  of containers the enemy chooses) is at least  $V'$ .

**Robust traveling salesman: the simple case** Given a (directed) multigraph with weighted edges, a budget, and a threshold  $t$ .

**Question:** For any choice of  $t$  edges destroyed by the adversary, will a path exist which starts and ends at the same vertex, includes every other vertex exactly once, and of which the total cost of edges is less than the budget?



## 4. A GENERAL APPROACH

### 4.1. NOTATION:

When  $\mathcal{S}$  is a set, we let  $\mathcal{S}^0 = \emptyset$ ,  $\mathcal{S}^1 = \mathcal{S}$ ,  $\mathcal{S}^i = \mathcal{S}^{i-1} \times \mathcal{S}$ , where  $i \geq 2$ .

### 4.2. PRELIMINARY:

Let  $\mathcal{P}$  be a set.  $\mathcal{A}_{\mathcal{P}} \subset 2^{\mathcal{P}}$  is an **adversary structure** if for each element  $B$  of  $\mathcal{A}_{\mathcal{P}}$  each subset of  $B$  also belongs to  $\mathcal{A}_{\mathcal{P}}$ , and  $\mathcal{P} \notin \mathcal{A}_{\mathcal{P}}$ . (Hirt-Maurer 2000)

**Example:** Let  $\mathcal{P} = \{1, 2, 3, 4\}$ . An example of an adversary structure:  $\{\emptyset, \{1\}, \{2\}, \{1, 2\}, \{3\}, \{1, 3\}, \{4\}\}$ .

**Note:** The adversary structure allows to model the case the adversary in the robust traveling salesman problem corresponds to a strike of an airline.



**Definition 1.** Let  $f$  be a Boolean function. We call  $f$  **properly deletion defined** if its domain can be written as  $G \times H$ , where  $G$  is the Cartesian product of a finite number ( $\ell$ ) of sets  $G_i$ , where  $G_i = \mathcal{S}_i \cup \mathcal{S}_i^2 \cup \mathcal{S}_i^3 \dots \cup \mathcal{S}_i^j \cup \dots$ , where  $\mathcal{S}_i$  is a set ( $0 \leq i \leq \ell$ ) and  $H$  is the Cartesian product of finitely many  $\ell'$  sets  $\mathcal{S}'_i$ .

### 4.3. INSTANCE:

**Definition 2.** An instance  $x$  corresponds to  $x = (a, b)$  where  $a \in G$ ,  $b \in H$ , where  $a = (a_1, a_2, \dots, a_\ell)$  in which  $a_i \in \mathcal{S}_i^{k_i}$ , where all  $k_i$  are finite. To such an instance will correspond an adversary structure  $\mathcal{A}_{\mathcal{P}}$ , where

$$\mathcal{P} = \{(i, j) \mid 1 \leq i \leq \ell \quad \text{and} \quad 1 \leq j \leq k_i\},$$

i.e. the  $i$  coordinate corresponds to the specification of set  $\mathcal{S}_i$  and  $j$  corresponds with a number between 1 and  $k_i$ . Since  $\mathcal{P}$  is naturally defined by the domain of  $x$ , we will also use the notations  $\mathcal{A}_x$  and  $\mathcal{P}_x$ .

## 4.4. ROBUST OPERATION:

**Definition 3.** Assume that the operational research problem  $P$  is specified by a Boolean function  $f$  which is properly deletion defined. We say that an instance  $x = (a, b)$  is robust if for **each**  $B \in \mathcal{A}_x$ ,  $f(x'_B)$  is TRUE, where  $x'_B = (a'_B, b)$  in which  $a'_B$  is identical to  $a$ , **except** that the coordinates of  $a'_B$  specified by  $B$  are removed.

We call the corresponding computational problem, the robust  $P$  problem.

## 5. AN ILLUSTRATION

Let us reconsider the simple robust multiple knapsack.

### 5.1. ILLUSTRATING THE NOTATION

Let  $\mathcal{S}_1$  corresponds to the set used to express sizes. So,  $\mathcal{S}_1^m$  is used to express the size of the  $m$  containers. Let  $\mathcal{S}_2 = \mathcal{T} \times \mathcal{S}_1$ , where  $\mathcal{T}$  is the set that expresses values. So,  $\mathcal{S}_2^n$  is used to express the (value, size) of each of the  $n$  items. Finally  $\mathcal{S}'_1 = \mathcal{T}$  and  $V' \in \mathcal{S}'_1$ .

So, an instance  $x$  of the multiple knapsack has the form:

$$x = ( ( (S_1, S_2, \dots, S_m), ( (s_1, v_1), (s_2, v_2), \dots, (s_n, v_n) ) ), V' ).$$

Then  $\mathcal{P}_x = \{(1, 1), (1, 2), \dots, (1, m), (2, 1), (2, 2), \dots, (2, n)\}$ . Since  $\mathcal{A}_x$  is an element of the powerset of  $\mathcal{P}_x$ , it does specify which

- **containers the enemy can destroy**, as well as which
- **items** can disappear.

In the robust multiple knapsack only containers could be destroyed, but items never disappeared. So, we call this the **general robust multiple knapsack**.

## 5.2. A SCENARIO:

- the  $m$  so called “containers” are ships.
- The  $n$  items are containers that are transported by trucks to the ships.

Due to the threat of the adversary, some of the  $n$  trucks on their way to the ships, as well as some of the  $m$  ships could be destroyed by the adversary.

## Note:

Evidently since  $\mathcal{A}_x$  is an element of the powerset of  $\mathcal{P}_x$ , we can also model the simple robust multiple knapsack as a special case of the general one, by choosing an appropriate  $\mathcal{A}_x$ .

## 6. VARIANTS

In **Robust Traveling Salesman**: we assumed the adversary strikes **before** the salesman travels. So the adversary first announces which airlines will not fly, and then the task is to choose the shortest path.

Let us now imagine the adversary chooses which airlines strikes after the salesman started to travel. A problem that may occur is that the salesman is stuck.