Robust Operations

Yvo Desmedt*

Department of Computer Science, University College London, United Kingdom

Abstract *Operations under malicious attack are usually studied in a very narrow context. The typical example is in the context of point-to-point communication networks in a graph. The question is whether after k nodes have been destroyed, each remaining node can continue communicating to any remaining one. The topic was generalized to (1) secure distributed computation (secure multiparty computation); (2) secure communication, further generalized to the case of partial broadcast; (3) AND/OR graphs requiring minimal flows after the attack. In this paper we broaden these topics to robust operations in general.*

We give several examples, such as the robust multiple knapsack*, and the* robust traveling salesman*. We give a general definition.*

Keywords operational research, security, information security, adversary structure, robust control, critical infrastructures

1 Introduction

Operations under malicious attack have been studied in narrow contexts. One finds typical examples in communication and computer security, although there has been some work on limited generalizations. Evidently, there appears to be a relationship with the area of reliability. However, there is a major difference. In the case of reliability theory the failure is assumed to be ergodic. However, assuming ergodicity makes little sense for a malicious attack.

One of the first problems studied originates from the problem of reliable communication in a network. Imagine an adversary destroys k nodes. An adversary will fail to disrupt communication if the network is $k+1$-connected, i.e. any node can still communicate with any other (see also [17]). If k nodes are Byzantine (i.e. they can modify data they are supposed to just forward, stop interacting, etc.) then one needs $2k+1$ vertex disjoint paths between sender and receiver to achieve authentic (i.e. guarantee that the received message is the sent one and coming from

*A part of this research was done while visiting Macquarie University, Sydney, Australia, supported by ARC grant DP0344444 and the Information Security Institute at Queensland University of Technology, Brisbane Australia, as QUT Visiting Fellow. This research was partially funded by NSF grant CCR-0209092. The author is BT chair of information security. He is also a courtesy professor at Florida State University.

the claimed source) and robust (reliable) message transmission without prior key exchange between sender and receiver [17, 10].

This topic was generalized to secure distributed computation (called secure multiparty computation by the cryptographic community). In this, a set of mutually untrusted players wants to compute an arbitrary function f of their private inputs, while satisfying two major concerns, being:

correctness, i.e. the output of f is correct using the input of all honest players (and replacing the one of dishonest ones by randomness),

privacy, i.e. the privacy of the players inputs will be guaranteed, even if some of the players are corrupted by an adversary and are malicious (Byzantine).

This topic was introduced by Goldreich-Micali-Wigderson [16] (see also e.g. [21, 4, 1]).

In multiparty computation (without assuming computational assumptions): one assumed a complete graph for private and robust (reliable) communication. Dolev-Dwork-Waarts-Yung [11] challenged this. They studied two cases:

1. all communication links (edges in the graph) are two-way communications,
2. all communication links are one-way communication, and there is no feedback.

They were able to demonstrate that a complete (directed) graph is not required. Desmedt and Wang [9] observed that this is not the most general case since there could be feedback channels and demonstrated that this is indeed a new case. This problem of private and robust communication was further generalized to include the broadcast case [14, 6]. Special cases were studied in [13, 20] (see also [9]).

A first approach to broaden the research on robust computer networks to include non-information infrastructures was made in [3]. We could view it as one the first research papers towards robust operations. It suggested to use an AND/OR graph, the AND to model dependencies (as in a PERT graph) and the OR to model redundancies. Desmedt-Wang [8] (see also [5]) added flows to above model. The main (from our viewpoint) question studied was whether an adversary when destroying a number of nodes bounded above by a threshold in such an AND/OR graph, can reduce the delivered flow to a below a critical level. This issue is important when one is concerned whether a terrorist with limited resources can succeed in reducing water/food/fuel/electricity distribution to such a low level that people will die or the economy will suffer to a point beyond return. A macro approach was proposed by Desmedt-Burmester-Wang using an economics model in [7].

A topic developed independently is the one of robust control. Informally, one could view robust control as a part of control theory, which deals with control systems that are tolerant to changes in the environment, in the system parameters and in the system [12].

From these viewpoints one can view that robustness guarantees that desired properties remain, even if insiders behave dishonestly. It is this viewpoint that is the driving force of this paper.

This paper is organized as following. In Section 2 we introduce some examples to explain the ideas behind robust operations. We then generalize these to explain the concept of adversary structure in this context (see Section 3). We use these examples to define robust operations in general (see Section 4). Finally in Section 5 we conclude and present open problems.

2 Simple examples

To motivate the general definition, we introduce the issue of robust operations using some examples.

2.1 Robust multiple knapsack: the simple case

We first of all remind the reader about the *multiple knapsack* problem, also called *multiple loading problem*. It is the problem of choosing some of n items, each having its size s_i and value v_i, to be loaded into m distinct containers (or trucks), such that the total value of the selected items is maximized, without exceeding the capacity S_j of each of the containers j (or trucks).

We now define the basic *robust multiple knapsack*.

Definition 1 The adversary can destroy up to t containers (or trucks), specified by a subset B of the m containers ($|B| \leq t$). The *multiple knapsack* is *robust* if the transported value remaining after the attack (regardless which subset B the adversary chooses) is at least V'.

We briefly discuss the problem informally. When loading the containers, one cannot put the most valuable items in one container. Indeed if the adversary targets this container, then the total value transported using the remaining containers may be too low.

2.2 Robust traveling salesman: the simple case

Definition 2 Given a (directed) multigraph with weighted edges, a budget, and a threshold t. We call the weight of the edge, its cost. **Question:** For any choice of up to 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?

We now explain in the next section how to generalize these two examples to what is called general adversary structures.

3 More general examples

3.1 Introduction

The above adversarial situations may be too symmetric or too far from realistic. We explain this for both examples.

Let us start with the basic robust multiple knapsack. What would happen if *some* containers are escorted by security guards, or are armored. The adversary may not have the resources to destroy t armored containers. However, he might be able to destroy, let say 1 armored and t' unarmored.

Let us now focus on the robust traveling salesman. If the transportation mode is based on airplanes, then if the adversary is a terrorist organization, the scenario of taking out edges, except when using hijacking, may not be possible. However, the adversary does not need to be restricted to a terrorism scenario. Consider the case of a salesman that is traveling during a strike of pilots of some airline company. Another possibility is that the personnel of two or more airline companies are striking.

3.2 Background

We will use the concept of adversary structure which was introduced in the area of cryptography [18] (see also [19]).

Definition 3 Let $\mathcal{P}$ be a set. A subset $\Gamma_{\mathcal{P}}$ of the powerset $2^{\mathcal{P}}$ of $\mathcal{P}$ is called an access structure on $\mathcal{P}$ [19]. It is monotone if and only if $\emptyset \notin \Gamma_{\mathcal{P}}$ and supersets of elements $\Gamma_{\mathcal{P}}$ also belong to $\Gamma_{\mathcal{P}}$, i.e. formally we require that if $A \in \Gamma_{\mathcal{P}}$ and $A \subseteq A' \subseteq \mathcal{P}$, then $A' \in \Gamma_{\mathcal{P}}$. We call $\mathcal{A}_{\mathcal{P}} \subset 2^{\mathcal{P}}$ an adversary structure [18] on $\mathcal{P}$ if its complement, i.e., $\mathcal{A}^c_{\mathcal{P}} = 2^{\mathcal{P}} \setminus \mathcal{A}_{\mathcal{P}}$ is a monotone access structure.

3.3 A first generalization

In the examples of Section 2 the adversary could destroy either up to t containers (or trucks) or up to t edges. In the generalization, the set of containers (or the set of edges) will correspond to the set $\mathcal{P}$. One then has an adversary structure $\mathcal{A}_{\mathcal{P}}$ defined over $\mathcal{P}$. So, $\mathcal{A}_{\mathcal{P}}$ can be viewed as a list of subsets of the the set of containers (edges) from which the adversary can choose one. Obviously a special case of such an adversary structure is the one defined by the threshold, formally,

$$\mathcal{A}_{\mathcal{P}} = \{B \mid B \subset \mathcal{P} \quad \text{and} \quad |B| \leq t\}.$$

Using an adversary structure we can model the fact that some containers (or trucks) are armored and harder to take out. Similarly the adversary structure allows to model the list of all flight numbers from 1, 2 or up to t' airlines. One way to visualize this is to color the edges corresponding with flights of the same airline with the same color (see also [2]). If there is a strike of the personnel of t' airlines, this corresponds to removing the edges of these colored in one of these t' colors.

4 Robust operations

The goal of this section is to define robust operations in its generality, i.e. for any operational research problem. This is not straightforward for the reasons we now explain.

4.1 A first attempt

Using theoretical computer science terminology (see e.g [15]), we can split the operational research problems into decisional and search problems. In the first the question is whether something is possible or not, while in the second one wants a more complex reply. The last one are used to study optimization problems.

For simplicity we start with discussing how to define robust decisional problems. When the input is defined by elements of sets $\mathcal{S}_1$, $\mathcal{S}_2$, ..., $\mathcal{S}_\ell$ (some of these sets could be equal) one could try to define an access structure on $\mathcal{P} = \{1, 2, \ldots, \ell\}$. If the adversary removes a set $B = \{i_1, i_2, \ldots, i_{|B|}\} \subset \mathcal{P}$ the goal of the robust operation is that the same question is asked, but then the inputs corresponding to the sets $\mathcal{S}_{i_1}$, $\mathcal{S}_{i_2}$, ..., $\mathcal{S}_{i_\ell}$ are removed. The problem with this approach is that the original question no longer makes sense. Indeed, consider the decisional multiple knapsack and suppose that the minimal value V' is removed from the input. It is clear that the original question becomes void. To address this we split up the input, as we now describe.

4.2 Robust operations: the decisional case

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 $i \geq 2$.

Definition 4 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 (ℓ) of sets G_i, where $G_i = \mathcal{S}_i \cup \mathcal{S}_i^2 \cup \mathcal{S}_i^3 \ldots \cup \mathcal{S}_i^j \cup \ldots$, where $\mathcal{S}_i$ is a set ($0 \leq i \leq \ell$) and H is the Cartesian product of finitely many ℓ' sets $\mathcal{S}'_i$.

Definition 5 An instance x corresponds to $x = (a, b)$ where $a \in G$, $b \in H$, where $a = (a_1, a_2, \ldots, a_\ell)$ 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 $\mathcal{P}_x$.

Note that *some* families of adversary structures $\{\mathcal{A}_\mathcal{P}\}$ can be described in polynomial size in function of $|\mathcal{P}|$.

We now define a robust version of operational research problems.

Definition 6 Assume that the operational research problem P is specified by a Boolean function f which is properly deletion defined. We say that an instance $x = (a, b)$ (using the same notations as in Definition 5) 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.

4.3 An illustration

To better understand above definition, let us focus on an example.

Let us reconsider the robust multiple knapsack. In fact we will see that our discussion leads us naturally to what we call the *general robust multiple knapsack*. We use the notations from Section 2.1. Obviously $\ell \leq 3$. We now argue that $\ell = 2$ makes the most sense.

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:

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

Then $\mathcal{P}_x = \{(1,1), (1,2), \ldots, (1,m), (2,1), (2,2), \ldots, (2,n)\}$. Since $\mathcal{A}_x$ is an element of the powerset of $\mathcal{P}_x$, it does not only specify which

1. containers the enemy can destroy, these are the elements in $\mathcal{P}$ of the form $(1, \cdot)$, but also which
2. items can disappear, these are the elements in $\mathcal{P}$ of the form $(2, \cdot)$. Since $\mathcal{S}_2 = \mathcal{T} \times \mathcal{S}_1$, both the value as well as the size of each item that disappears, are removed from x to make x'_B.

In the robust multiple knapsack only containers could be destroyed, but items never disappeared. One could wonder whether the generalized scenario is realistic.

We now explain such a scenario. Imagine that the m so called "containers" are actually 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.

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 $\mathcal{A}_x$.

4.4 Technical comments

We note that V' in the general robust multiple knapsack cannot be removed by the adversary since it belongs to $\mathcal{S}'_1$, which solves the problem we encountered

in Section 4.1, when we defined $\mathcal{P}$ over all coordinates of the input. Evidently we could also have achieved this by restricting $\mathcal{A}_x$ by stating that these special inputs are never removed. However, the last approach would not allow a natural framework for defining robust operations in its generality.

For a given instance x, and for a choice B of the adversary, where $B \in \mathcal{A}_x$, one could view the removal as a mapping from

$$((\mathcal{S}_1^{k_1}, \mathcal{S}_2^{k_2}, \ldots, \mathcal{S}_\ell^{k_\ell}), (\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_{\ell'}))$$

to

$$((\mathcal{S}_1^{k_1-b_1}, \mathcal{S}_2^{k_2-b_2}, \ldots, \mathcal{S}_\ell^{k_\ell-b_\ell}), (\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_{\ell'})),$$

where $0 \leq b_i \leq k_i$ and $|B| = \sum_i b_i$.

Evidently one would not have obtained the general robust multiple knapsack if one would had started from the ordinary knapsack. Only items could then be destroyed, but not the container. One could view the single knapsack as an "hidden variable." So when using the above, one may want to first identify hidden variables, then make a "multiple" version of the original problem. The multiple knapsack is such a variant of the original one. Note that the robust traveling salesman is defined over a *multi*graph to deal exactly with this issue.

4.5 The search problem case

The search based operational research problems seems not so interesting. Indeed the question then is to solve the original search problem after the adversary removed some of the inputs.

4.6 When does the adversary strike?

Let us rediscuss the decisional robust traveling salesman. We have silently assumed that 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. Indeed, it could happen that all cities that have edges adjacent with his current location have already been visited by him. Worse, if he had done his travel differently, he might have avoided this problem.

5 Conclusion and open problems

The topic introduces almost[1] as many new operational research problems as there currently are. Evidently to robust operational research correspond a few natural open questions, such as:

[1]The reason it is not the double is that a few problems, such as the one on deciding whether a graph is sufficiently connected, already deal with robustness.

- for which of these can one find efficient algorithms?
- which of these are hard, and how hard?
- when they are hard, are there efficient heuristic algorithms?

Acknowledgment

The author thanks Yongge Wang for the many discussions on how to model our critical infrastructures.

References

[1] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM Symp. Theory of Computing, STOC*, pp. 1–10, May 2–4, 1988.

[2] M. Burmester and Y. G. Desmedt. Is hierarchical public-key certification the next target for hackers? *Communications of the ACM*, 47(8), pp. 68–74, August 2004.

[3] M. Burmester, Y. Desmedt, and Y. Wang. Using approximation hardness to achieve dependable computation. In M. Luby, J. Rolim, and M. Serna, editors, *Randomization and Approximation Techniques in Computer Science, Proceedings (Lecture Notes in Computer Science 1518)*, pp. 172–186. Springer-Verlag, October, 8–10 1998. Barcelona, Spain.

[4] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM Symp. Theory of Computing, STOC*, pp. 11–19, May 2–4, 1988.

[5] Y. Desmedt and Y. Wang. Analyzing vulnerabilities of critical infrastructures using flows and critical vertices in and/or graphs. *International Journal of Foundations of Computer Science*, 15(1), pp. 107–125, February 2004.

[6] Y. Desmedt, Y. Wang, R. Safavi-Naini, and H. Wang. Radio networks with reliable communication, 2005. Kunming, Yunnan China, August 16-19, 2005.

[7] Y. Desmedt, M. Burmester, and Y. Wang. Using economics to model threats and security in distributed computing. Workshop on Economics and Information Security, Berkeley, May 16-17, 2002, `http://www.sims.berkeley.edu/resources/affiliates/workshops/econsecurity/econws/33.ps`.

[8] Y. Desmedt and Y. Wang. Maximum flows and critical vertices in and/or graphs. In O. H. Ibarra and L. Zhang, editors, *Computing and Combinatorics, 8th Annual International Conference, COCOON 2002 (Lecture Notes in Computer Science 2387)*, pp. 238–248. Springer-Verlag, 2002. Singapore, August 15-17.

[9] Y. Desmedt and Y. Wang. Perfectly secure message transmission revisited. In L. Knudsen, editor, *Advances in Cryptology — Eurocrypt 2002, Proceedings (Lecture Notes in Computer Science 2332)*, pp. 502–517. Springer-Verlag, 2002. Amsterdam, The Netherlands, April 28–May 2.

[10] D. Dolev. The Byzantine generals strike again. *Journal of Algorithms*, 3, pp. 14–30, 1982.

[11] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *Journal of the ACM*, 40(1), pp. 17–47, January 1993.

[12] G. E. Dullerud and F. Paganini. *A Course in Robust Control Theory*. Springer-Verlag, New York, 2000.

[13] M. Franklin and R. Wright. Secure communication in minimal connectivity models. In K. Nyberg, editor, *Advances in Cryptology — Eurocrypt '98, Proceedings (Lecture Notes in Computer Science 1403)*, pp. 346–360. Springer-Verlag, 1998. Espoo, Finland, May 31–June 4.

[14] M. K. Franklin and M. Yung. Secure hypergraphs: Privacy from partial broadcast. In *Proceedings of the twenty seventh annual ACM Symp. Theory of Computing, STOC*, pp. 36–44, 1995.

[15] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco, 1979.

[16] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *27th Annual Symp. on Foundations of Computer Science (FOCS)*, pp. 174–187. IEEE Computer Society Press, 1986. Toronto, Ontario, Canada, October 27–29, 1986.

[17] V. Hadzilacos. *Issues of Fault Tolerance in Concurrent Computations*. PhD thesis, Harvard University, Cambridge, Massachusetts, 1984.

[18] M. Hirt and U. Maurer. Player simulation and general adversary structures in perfect multiparty computation. *Journal of Cryptology*, 13(1), pp. 31–60, 2000.

[19] M. Ito, A. Saito, and T. Nishizeki. Secret sharing schemes realizing general access structures. In *Proc. IEEE Global Telecommunications Conf., Globecom'87*, pp. 99–102. IEEE Communications Soc. Press, 1987.

[20] Y. Wang and Y. Desmedt. Secure communication in broadcast channels. *Journal of Cryptology*, 14(2), pp. 121–135, 2001.

[21] A. C. Yao. How to generate and exchange secrets. In *27th Annual Symp. on Foundations of Computer Science (FOCS)*, pp. 162–167. IEEE Computer Society Press, 1986. Toronto, Ontario, Canada, October 27–29, 1986.