

Constraint isomorphism and correction algorithms for violations

Toshinori Munakata^{1,*} and Adam Fadlalla^{1,**}

¹ Computer and Information Science Dept., Cleveland State University, USA.

When a problem can be characterized by different parameter sets, constraints imposed on one parameter set must be consistent with, or isomorphic to, constraints imposed on other parameter sets. When mapping among the parameter sets is not so visibly obvious, one might overlook consistency violations, which have occurred in the past. This paper examines such violations and possible correction algorithms.

© 2007 WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim

We address a basic principle on dealing with multiple parameter sets: *When a set of constraints is imposed in terms of one parameter set, the same constraints must be observed by another set, and vice versa.* We call such consistency among parameter sets *constraint isomorphism* [1]. This issue can occur in many application domains such as science, engineering and operations research. A set of parameters that represents the characteristics of data may be associated with an application problem. Additional parameters are often introduced to represent the characteristics of the problems. Reynolds number in fluid dynamics and traffic density in queuing theory are such examples.

The issue discussed here may be best understood by a specific case study: the total tardiness problem, an NP-complete job scheduling problem. In this problem, two parameter sets are commonly employed. For easy reference, we call one set “data parameters” and the other “characteristic parameters.” Processing time and due date for each individual job are data parameters, and the tardiness factor, τ , and relative range of due dates, δ , are characteristic parameters. For the simplest, single machine model, each job is processed by the single machine at a time. n jobs wait for processing at time $t = 0$, and they have *processing times* p_1, p_2, \dots, p_n , and *due dates* d_1, d_2, \dots, d_n , respectively. The objective of the problem is to determine the order of jobs to be processed to minimize the *total tardiness*.

The total tardiness problem has been extensively researched, employing various algorithms and heuristics. Since the complexities of exact algorithms to solve the total tardiness problem are exponential, simulations based on randomly generated data are typically performed to illustrate the performance of proposed algorithms and heuristics. A common scenario is that each of the τ and δ values is selected from a set of numbers, typically $0 \leq \tau, \delta \leq 1$. This domain is illustrated as the square in Fig. 1 (a). It has been shown that some of these selected pairs yield two kinds of violations: Type 1: Negative due date. The due date for every job must be non-negative, that is, $d_i \geq 0$. Type 2: Due date < processing time. For every job i , $d_i \geq p_i$ should hold. Fig. 1 (b) plots the (D_L, D_U) space, where D_L and D_U are the lower and upper bounds of due dates, respectively, and demonstrates how Type 1 violations can occur. The square in Fig. 1 (a) corresponds to the rectangle in Fig. 1 (b), which includes a region of negative due dates. As this case study illustrates, when mapping among the parameter sets is not so evident, one might overlook consistency violations, which have occurred in many published simulations. Constraint isomorphism is important in many situations, including generation of random data for simulation of stochastic problems.

Detecting inconsistencies among parameter sets and constraint violations as discussed above is one issue. Other important issues are how to prevent such violations and how to correct them if they occur. These issues are easier to deal with for Type 1 violations than for Type 2 violations, since the former involves only one variable, d_i , whereas the latter involves two variables, d_i and p_i . To deal with Type 2 violations, we have developed several algorithms. They include: discard-and-replace, modified probability distributions, and data-interchanging. In these algorithms, we assume that p_i is generated first, then d_i , as suggested by previous work. The general idea of the last algorithm is: Perform data interchanging starting from the most local data level to resolve violations; if not resolved, we extend the data level toward a more global level, until all violations are resolved.

Numerical experiments are performed to verify that the algorithm works. The upper bound of the computational complexity of this algorithm is $O(mn \log(mn))$, where m is the number of problems in the entire problem set under consideration.

* E-mail: munakata@cis.csuohio.edu

** E-mail: fadlalla@cis.csuohio.edu

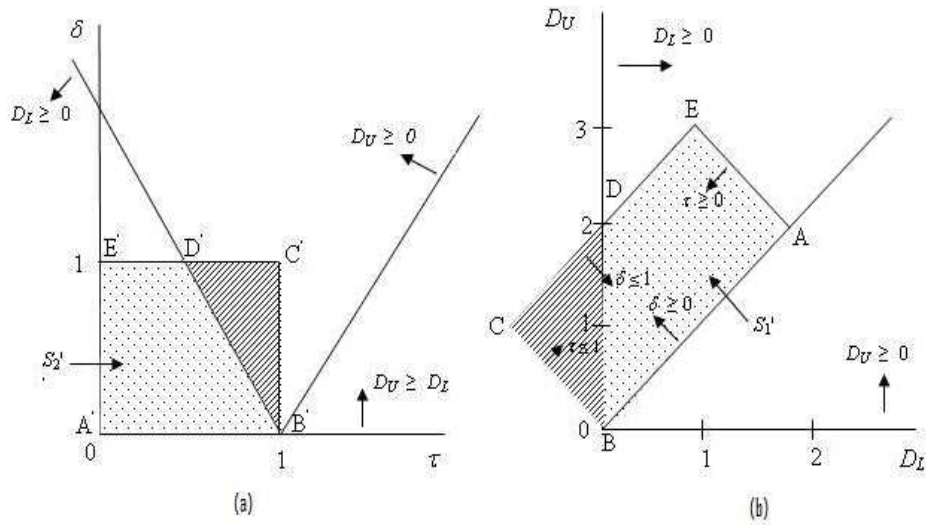


Fig. 1 (a) The domain of $0 \leq \tau \leq 1$ and $0 \leq \delta \leq 1$ on the (τ, δ) space. (b) The corresponding domain of Fig. (a) on the (D_L, D_U) space.

References

[1] Munakata, T. and Fadlalla, A. Constraint isomorphism and the generation of stochastic data, *IIE Transactions*, 38 (2006), 437-444.