Constraint isomorphism and the generation of stochastic data

TOSHINORI MUNAKATA* and ADAM FADLALLA

Computer and Information Science Department, Cleveland State University, Cleveland, OH 44115, USA
E-mail: munakata@cis.csuohio.edu or fadlalla@cis.csuohio.edu

Received August 2002 and accepted September 2004

We consider the problem of generating random data under constraints that are expressed in terms of different parameter sets. These constraints must be consistent between the parameter sets. However, this requirement of constraint consistency has to date not received much attention in the literature. The major objective of this article is to propose a formal concept called constraint isomorphism to detect and help avoid inconsistencies between the constraints. The method presented here can be used as a verification technique for random-data generation. As a case study, we illustrate our methodology on the total-tardiness problem: a NP-hard job scheduling problem. Since generating random data under constraints is an extremely common problem, especially in the simulation arena, the technique has a wide spectrum of potential applications.

1. Introduction

The generation of stochastic data from a probability distribution is a standard procedure in the modeling and simulation of complex systems. This article deals with the question of handling constraints on the underlying parameters of the generated random variables or on the random variables themselves. For example, suppose that \( p_i \) is the random variable that denotes the processing time for job number \( i \). If \( p_i \) is uniform on \( p_L \) and \( p_U \), then obvious constraints on the parameters are \( p_L \geq 0 \) and \( p_U \geq p_L \). These conditions become indirect constraints on the associated random variable(s). In addition, constraints can also be imposed directly on random variables as, e.g., \( p_i \geq 0 \) or \( d_i \geq p_i \), where \( d_i \) is the random variable that corresponds to the due date for job \( i \).

In this article we define two types of parameters for easier discussion. There are often natural parameters associated with the data that we call “data” parameters. Furthermore, it is common practice to introduce additional “characteristic” parameters in many application problems. Typically, the data parameters are directly related to the random data, whereas the characteristic parameters are introduced to better represent the characteristics of the overall problem. The terms data and characteristic parameters are for our convenience and are not very rigid. As a rule of thumb, data parameters may be defined directly or in at most one round of arithmetic operations such as summations of the random data. The lower and upper bounds and the mean of a certain quantity are in this category. On the other hand, characteristic parameters have two basic features: (i) they are indirectly defined in terms of data parameters; and (ii) they are introduced to characterize some aspect of the problem, such as the difficulty of the problem. Although here we use two parameter sets, one for grouping data parameters and the other for grouping characteristic parameters, in some cases more than two parameter sets may be associated with the generated data, in which case one could simply call the sets parameter set 1, 2, and so on.

In the job scheduling problem that will be discussed in Section 3, each job has a processing time and a due date that are randomly selected based on some probability distribution. The lower and upper bounds for the processing times are data parameters. The characteristic parameters would include the tardiness factor which represents the ratio of the average number of jobs that do not finish on time to the total number of jobs. This characteristic parameter is derived from the data parameters, and represents a characteristic (i.e., the difficulty) of the overall tardiness problem.

To illustrate further, consider a queuing problem (Hillier and Lieberman, 2002). Customers arrive at a service station of \( m \) servers at rate \( \lambda \) and are served at rate \( \mu \). These parameters \( m, \lambda, \) and \( \mu \) can be considered as data parameters since they are more or less directly associated with the randomly generated data. In addition, the traffic intensity \( \rho = \lambda/(m\mu) \) is the steady-state fraction of the time that the server is busy, and characterizes the difficulty of the problem: the higher the \( \rho \) value, the harder the problem. Thus, the parameter \( \rho \) can be considered as a characteristic parameter, since it is...
derived indirectly from data parameters and can be used to characterize the problem.

When constraints are imposed on different parameter sets, the constraints on one parameter set must be consistent with the constraints on the other parameter set so that the randomly generated data are consistent. If the constraints imposed on the characteristic parameters are inconsistent with those imposed on the data parameters, then the data sets generated under those conditions will violate one of the constraint sets. In the job scheduling problem, if each processing time is always non-negative for data generated under constraints on the data parameters whereas it is not for data generated under constraints on the characteristic parameters, then this leads to an inconsistency in the generated data sets. We have found that some past research has failed to address these consistency issues (see Section 4). Recently Hall and Posner (2001) raised a similar question regarding data generation schemes for computational experiments. They proposed several principles for data generation and suggested several desirable properties for the data generation schemes. In this article, we give a formal presentation on how to detect and then avoid such violations by introducing the concept of constraint isomorphism.

We begin in the following section by introducing the concept of sample and parameter spaces and define constraint isomorphism, the core idea of this article. In Section 3 we present a case study (the total-tardiness problem) and describe previous work. Section 4 considers two types of constraints for this problem. Type-1 constraints are on individual variables that require non-negative due dates and are discussed in Section 5. Type-2 constraints are intervariable constraints that require the due dates to be greater than or equal to the processing time for every job and are discussed in Section 6. Section 7 provides a summary and concluding remarks.

2. Sample and parameter spaces and constraint isomorphism

In this section we discuss sample and parameter spaces and define the concept of constraint isomorphism. The presentation here is somewhat general, but it will be more specific in Section 3.2 where these concepts are applied to a concrete problem. Consider the simple case in which two values are associated with each sample so that the sample space is two-dimensional. Generated samples will be scattered in this two-dimensional space represented by points, i.e., as $P_{11}, P_{12}, \ldots, P_{2n}$. If two parameters are associated with the random samples, then we can think of an abstract two-dimensional parameter space. A particular point in this parameter space corresponds to a specific pair of parameter values, e.g., $\text{para}_1 = 0, \text{para}_2 = 10$. A sample space that corresponds to this particular point in the parameter space would contain samples (realizations) that correspond to the parameter values.

More generally, we consider a problem where $n$ samples (data points) are randomly generated, and each sample is represented by $k$ variables, $x_1, \ldots, x_k$. For a simple queuing problem, $n$ can represent the number of units that arrive at a service station, $x_1$ the arrival time and $x_2$ the service time for each unit. Then the $n$ samples can be represented by a $n \times k$ matrix:

$$
X = \begin{bmatrix}
    x_{11} & x_{12} & \cdots & x_{1k} \\
    \vdots & \vdots & & \vdots \\
    x_{n1} & x_{n2} & \cdots & x_{nk}
\end{bmatrix},
$$

where $x_{ij}$ represents the value of the $i$th sample of variable $x_j$. So, for example, $x_{11}$ could be the $1$th customer's arrival time.

Let the set of constraints to be imposed on $X$ be $C_0(X)$. For a specific problem, if every $x_{ij}$ must be non-negative, then the condition $x_{ij} \geq 0, i = 1, \ldots, n, j = 1, \ldots, k$, is a constraint, that is, an element of $C_0(X)$. A constraint can be inter-variable in nature; for example, for specific values of $j$ and $j'$, we might require $x_{ij} \geq x_{ij'}, i = 1, \ldots, n$. Every constraint in $C_0(X)$ must be satisfied, and conversely, if every constraint in $C_0(X)$ is satisfied, then the generated random data is feasible. Using our terminology, let $Q = (q_1, \ldots, q_n)$ be the set of data parameters, and $\Pi = (\pi_1, \ldots, \pi_k)$ be the set of characteristic parameters. Furthermore, let the set of constraints on $Q$ be $C_1(Q)$, and the set of constraints on $\Pi$ be $C_2(\Pi)$.

We can consider abstract $k$, $\alpha$, and $\beta$-dimensional spaces defined on $X$, $Q$ and $\Pi$, respectively. The set of constraints for each space specifies a certain region or domain within the space. For $X$, $C_0(X)$ specifies a certain domain $S_0$ in the $k$-dimensional space. The $n$ samples that are generated randomly for a specific run and are represented as $n$ scattered points within this domain must be within $S_0$; otherwise some constraints are violated. Similarly, for $Q$ and $\Pi$, $C_1(Q)$ and $C_2(\Pi)$ specify their domains, $S_1$ and $S_2$, respectively. Parameters $q_1, \ldots, q_n$ and $\pi_1, \ldots, \pi_k$ must be confined within $S_1$ and $S_2$, respectively.

We can map $(\pi_1, \ldots, \pi_k)$ to $(q_1, \ldots, q_n)$, and vice versa. For example, given $S_2$, we can determine its corresponding domain in the space defined on $Q$. The intersection of this domain and $S_1$ is denoted as $S_0$. Obviously $S_0$ satisfies both $C_1(Q)$ and $C_2(\Pi)$. Similarly, given $S_1$, we can determine its corresponding domain in the space defined on $\Pi$. The intersection of this domain and $S_2$ is denoted as $S_0$ and it satisfies both $C_1(Q)$ and $C_2(\Pi)$. Let $\xi_1$ and $\xi_2$ be arbitrary points in $S_2$ and $S_1$, respectively. The two domains $S_1$ and $S_2$ are said to be constraint isomorphic (or simply isomorphic) if there is a one-to-one correspondence between $\xi_1$ in $S_1$ and $\xi_2$ in $S_2$. Constraint isomorphic domains $S_1$ and $S_2$ determined by the process described here are consistent with one another, i.e., a point in either one satisfies the set of constraints of the other one. It is possible to have mappings that are not one-to-one mappings but that are constraint consistent. Such cases are more complex and are not considered in this article.
In the following sections, we illustrate the concept of isomorphism with a case study of the total-tardiness problem; a well-known NP-hard job scheduling problem. In particular, we discuss how the spaces $X$, $Q$ and $\Pi$ can fit this application. We show how our approach can detect constraint violations in the context of the tardiness problem.

3. Case study: The total-tardiness problem

3.1. A general description of the total-tardiness problem

Although the tardiness problem is well known, we briefly describe it for the reader's convenience (French, 1983; Abdul-Razaq et al., 1990; Potts and Van Wassenhove, 1991; Koukamas, 1994; Baker, 1995). At time $t = 0$, $n$ jobs are waiting to be processed. In the simplest, single-machine model, one job is processed to completion by a single machine. The $n$ jobs require processing times $p_1, \ldots, p_n$, respectively. The $n$ jobs are also assigned due dates $d_1, \ldots, d_n$, respectively. These are absolute dates counted from time $t = 0$. The objective of the problem is to determine the order of jobs to be processed so as to minimize the total tardiness, which is the sum over all jobs of the number of days by which the completion time of each job exceeds its due date. In the following we list parameters and notation relevant to the tardiness problem.

In the parameter subscripts: “$s$” stands for set which is a parameter value that is either explicitly assigned a specific constant or can be derived from other preset parameters. We may omit the subscript “$s$” when it is clear from the context; “e” stands for expected and is the theoretical expected value for a data generation method; “a” stands for actual and is the actual (realized) value obtained from an experimental run. Also “L” and “U” stand for the lower and upper bound respectively.

Commonly used data parameters for the tardiness problem include: $p_L$ and $p_U$ which are the preset lower and upper bounds on the processing time; and $p_a$ which is the average processing time. There are three average processing times: $p_{v,s}, p_{v,e}$, and $p_{v,a}$. In much of the literature on the tardiness problem, the uniform distribution between $p_L$ and $p_U$ is assumed for processing time $p_i$ in which case the preset average processing time, $p_{v,s}$, is $(p_L + p_U)/2$. This value may or may not be the same as the expected average processing time, $p_{v,e}$, for a specific method, because the method may have different ($p_L$, $p_U$) or a different probability distribution (possibly due to inconsistent constraints). Both $p_{v,s}$ and $p_{v,e}$ are typically different from the actual (sample) average processing time, $p_{v,a}$, of a given run for a specific random generation method. The notations and properties for the processing time $p$ also apply for due date $d$. The uniform distribution between $d_L$ and $d_U$ is also commonly assumed for due date $d_i$.

Two characteristic parameters, $\tau$ and $\delta$, called the tardiness factor and relative range of due dates, respectively, are employed to represent the “difficulty” of a specific tardiness problem. The quantities $\tau$ and $\delta$ are defined as follows:

$$\tau = 1 - \frac{d_{u,s}}{np_{v,s}}, \quad 0 \leq \tau \leq 1, \quad (2)$$

$$\delta = \frac{d_{u} - d_{l}}{np_{v,s}}, \quad 0 \leq \delta \leq 1. \quad (3)$$

The tardiness factor, $\tau$, represents the ratio of the average number of jobs that do not finish on time to the total number of jobs. As a simple example, let us consider: an example with $n = 5$ jobs, with an average processing time, $p_{v,s}$ of 10 days, and an average due date, $d_{u,s}$ of 30 days. Furthermore, for simplicity assume that all jobs have the same processing time and due date. We can say that on the average, $d_{u,s}/p_{v,s} = 3$ jobs finish on time, each job spending 10 days processing time over the 30-day period. This yields $\tau = 0.4$. We see that the meaningful domain of $\tau$ is $0 \leq \tau \leq 1$. The higher the $\tau$ value, the harder it is to meet the due dates. The relative range of due dates, $\delta$, is a measure for the due date span, $(d_{u} - d_{l})$, over the total processing time, $(np_{v,s})$. Theoretically, $\delta$ can be greater than one, but such a case is considered to be trivial since none of the jobs will be tardy; so it is standard practice to use $0 \leq \delta \leq 1$.

In practice, $\tau$ and $\delta$ may be set to specific values (for example, $\tau = 0.8$ and $\delta = 0.2$); then $d_{l}$ and $d_{u}$ may be determined from Equations (2) and (3):

$$d_{l} = np_{v,s} \frac{2(1 - \tau) - \delta}{2}, \quad (4)$$

$$d_{u} = np_{v,s} \frac{2(1 - \tau) + \delta}{2}. \quad (5)$$

3.2. The tardiness problem in terms of sample and parameter spaces

The tardiness problem can be interpreted as a special case of the general description of sample and parameter spaces discussed in Section 2. For the sample space, we have two variables $x_1 = p$ and $x_2 = d$. Matrix $X$ given in Equation (1) reduces to a $n \times 2$ matrix as follows:

$$X = \begin{bmatrix}
\vdots & \vdots \\
p_1 d_1 \\
\vdots & \vdots \\
p_n d_n
\end{bmatrix}.$$ 

The set of constraints imposed on $X$, $C_0(X)$, can include $p_i, d_i \geq 0$, $p_L \leq p_i \leq p_U$, $d_L \leq d_i \leq d_U$, and $d_i \geq p_i$, for $i = 1, \ldots, n$. The last condition states that no resource would accept a job that takes more processing time than the due date. Each constraint except the last one is a constraint on a single variable, whereas the last one is an inter-variable constraint.

Let us assume that values of $n, p_{v,s}$ and $p_{v,e}$ are fixed; hence, $p_{v,s}$ is also fixed. Under this assumption, we can select our set of data parameters, $Q = (q_1, q_2) = (d_{l}, d_{u})$. Similarly, we can select our set of characteristic parameters, $\Pi = (\tau, \delta)$. The domain specified by the set of constraints
on \( Q, C_1(Q) \), must be isomorphic to the domain specified by the set of constraints on \( \Pi, C_2(\Pi) \). Particular forms of \( C_1(Q) \) and \( C_2(\Pi) \) will be discussed later.

3.3. Previous work on the total-tardiness problem

Extensive work has been performed on the tardiness problem, employing various methods, including several clever heuristics. Table 1 shows a selection of references that use heuristics for the single-machine model.

<table>
<thead>
<tr>
<th>Author(s)</th>
<th>( \tau )</th>
<th>( \delta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilkerson and Irvin (1971)</td>
<td>( \delta &lt; 0.85 )</td>
<td></td>
</tr>
<tr>
<td>Rinnooy Kan et al. (1975)</td>
<td>( 0.60 \leq \tau \leq 0.80 )</td>
<td></td>
</tr>
<tr>
<td>Srinivasan (1971)</td>
<td>( \tau = 0.65 )</td>
<td></td>
</tr>
<tr>
<td>Fisher (1976)</td>
<td>( 0.60 \leq \tau \leq 0.80 )</td>
<td>( \delta = 0.20 )</td>
</tr>
<tr>
<td>Potts and Van Wassenhove (1991)</td>
<td>( \tau = 0.60 )</td>
<td>( \delta = 0.20 )</td>
</tr>
</tbody>
</table>

Table 1. Problems in terms of \( \tau \) and \( \delta \), previously reported in the literature

4. Constraints for the problem

There are two variable constraints in the total-tardiness problem, that we label as Type 1 and Type 2:

Type 1: non-negative due date, \( d_i \geq 0 \), for \( 1 \leq i \leq n \);
Type 2: due date \( \geq \) processing time, i.e., \( d_i \geq p_i \) for \( 1 \leq i \leq n \).

Again, we note that a Type-1 constraint is applied to a single random variable, whereas a Type-2 constraint is an inter-variable constraint. These two constraints may not be obvious when the problem is expressed in terms of the two characteristic parameters, \( \tau \) and \( \delta \). The authors cited in Table 1 studied the performance of their heuristic algorithms by randomly generating test data. Additional articles that used the same type of test data include: Potts and Van Wassenhove (1982, 1985, 1987), Chambers et al. (1991), Holsenback and Russell (1992), Panwalker et al. (1993), Fadlalla et al. (1994, 1995), Koulamas (1994), Fadlalla and Evans (1995), Szwarc and Mukhopadhyay (1996), and Akturk and Yildirim (1999).

Table 2 shows the randomly generated data for example 1

<table>
<thead>
<tr>
<th>Job number</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proc. time, ( p_i )</td>
<td>55</td>
<td>85</td>
<td>24</td>
<td>8</td>
<td>23</td>
<td>64</td>
<td>93</td>
<td>68</td>
<td>19</td>
<td>76</td>
</tr>
<tr>
<td>Due date, ( d_i )</td>
<td>47</td>
<td>237</td>
<td>183</td>
<td>149</td>
<td>-21</td>
<td>97</td>
<td>52</td>
<td>0</td>
<td>79</td>
<td>111</td>
</tr>
<tr>
<td>Is ( d_i &lt; 0 )?</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Is ( d_i &lt; p_i )?</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

for each pair of \((\tau, \delta)\) values, yielding a total of \( 5 \times 25 = 125 \) problems.

This data generation procedure yields violations of both types of constraints mentioned above, depending on the values of \( \tau \), \( \delta \), \( n \), \( p_{\text{L}} \), and \( p_{\text{U}} \).

Example 1: A simple total tardiness run with \( n = 10 \), \( p_{\text{L}} = 1 \), \( p_{\text{U}} = 100 \), \( p_{\text{U}} = 50.5 \), \( \tau = 0.8 \), \( \delta = 0.6 \), \( d_{\text{L}} = -50.5 \), \( d_{\text{U}} = 252.5 \), and \( d_n = 101 \). Shown in Table 2 is a simple example, randomly generated following the procedure.

In this specific example, there is one Type 1 \((d_i < 0)\) and four Type 2 \((d_i < p_i)\) violations. We will show in Sections 5 and 6 that the probability for the Type-1 violation is 0.167 for this particular parameter value combination with the probability for the Type-2 violation being 0.333.

In the following sections, we discuss how to deal with Type-1 and Type-2 problems. Type-1 violations can be avoided by carefully selecting the values of the \( \tau \) and \( \delta \) parameters. On the other hand, Type-2 violations cannot and probably should not be avoided by simply selecting the values of the characteristic or data parameters. Type-2 violations must be dealt with by ensuring that \( d_i \geq p_i \) is true for each job \( i \).

5. Type-1 violations \((d_i < 0)\)

The dot-shaded area of Fig. 1(a) shows the domain \( S_1 \) in the \((D_{\text{L}}, D_{\text{U}})\) space, where \( D_{\text{L}} \) and \( D_{\text{U}} \) are the normalized versions of \( d_{\text{L}} \) and \( d_{\text{U}} \) obtained by setting \( D_{\text{L}} = d_{\text{L}}/(np_{\text{U}}/2) \) and \( D_{\text{U}} = d_{\text{U}}/(np_{\text{U}}/2) \). The constraints on \( D_{\text{L}} \) and \( D_{\text{U}} \) are \( D_{\text{L}} \geq 0 \) and \( D_{\text{U}} \geq D_{\text{L}} \). The dot-shaded area of Fig. 1(b) shows the domain \( S_2 \) in the \((\tau, \delta)\) space for the selection from the set \{0.2, 0.4, 0.6, 0.8, 1.0\} (the domain is extended to \( 0 \leq \tau \leq 1 \) and \( 0 \leq \delta \leq 1 \) for simplicity, but does not affect the essence of our discussion).

Using the normalized versions of Equations (4) and (5), we can map the values between \((D_{\text{L}}, D_{\text{U}})\) and \((\tau, \delta)\) (Fig. 2a and b). In the \((D_{\text{L}}, D_{\text{U}})\) space, it is clear that the line-shaded triangular area of BCD is invalid since it represents a negative \( D_{\text{L}} \). Having a negative value of \( D_{\text{L}} \) is the condition under which a Type-1 violation, \( d_i < 0 \), can occur. To correct this violation, the triangular area of BCD in the \((D_{\text{L}}, D_{\text{U}})\) space and the corresponding triangular area B′C′D′ in the \((\tau, \delta)\) space must be avoided. The remaining dot-shaded domains of \( S_1 \) and \( S_2 \) satisfy both \( S_1 \) and \( S_2 \) and...
Fig. 1. Two-dimensional spaces defined on normalized data parameters ($D_L$, $D_U$) and characteristic parameters ($\tau$, $\delta$). The two dotted domains are not isomorphic to one another, illustrating inconsistent constraint sets: (a) the dotted domain $S_1$ represents the feasible region for the constraints in the ($D_L$, $D_U$) space, $C_1(Q) = \{D_L \geq 0, D_U \geq 0, D_U \geq D_L\}$; and (b) the dotted domain $S_2$ represents the feasible region for the constraints in the ($\tau$, $\delta$) space, $C_2(\Omega) = \{0 \leq \tau \leq 1, 0 \leq \delta \leq 1\}$.

are isomorphic to one another, and thus yield no Type-1 violations. In the example of Section 4, any pair of ($\tau$, $\delta$) that falls in the triangle $B'CD'$ in Fig. 2(b) gives a negative $d_L$. Among the 25 pairs of ($\tau$, $\delta$), nine of them fall in the triangle and give negative $d_L$ values. The pair ($\tau$, $\delta$) = (0.8, 0.6) used in example 1 is one of these cases.

Given the probability distribution for $d_L$, where $d_L < 0$, the probability that a randomly selected job has a negative due date (a Type-1 violation) is the ratio of the area on $[d_L, 0]$ divided by the area on $[d_L, d_U]$. When $d_L$ is uniform, the probability is simply:

$$P_{d<0} = \frac{|d_L|}{d_U - d_L} = -\frac{d_L}{d_U - d_L} = -\frac{2(1 - \tau) - \delta}{2\delta}. \quad (6)$$

We note that $P_{d<0}$ is determined solely by $\tau$ and $\delta$, and does not depend on $n$, $p_L$, or $p_U$. For example, for ($\tau$, $\delta$) = (0.8, 0.6), we have $P_{d<0} = 0.167$ (c.f. example 1). The expected number of jobs with negative due dates is given by $nP_{d<0}$.

6. Type-2 violations ($d_i < p_i$)

Type-2 violations are a bit more difficult to deal with than Type-1 violations since the former constraint involves intervariable relations with two variables, $d_i$ and $p_i$. The following provides a simple illustration of Type-2 violations.

Example 2: Using the values of $n = 10$, $p_L = 1$, $p_U = 100$, $p_t = 50.5$, $\tau = 0.8$, $\delta = 0.3$, $d_L = 25.2$, $d_U = 176.8$, and $d_e = 101$ leads to the results listed in Table 3.

Fig. 2. Two-dimensional spaces illustrating isomorphic domains: (a) the domain ABCDE corresponds to $S_2$ in Fig. 1(b) and $A'B'C'D'E'$ in Fig. 2(b). The triangular subdomain BCD shows the invalid area where constraint $D_L \geq 0$ is violated. The dotted domain $S'_2$, the intersection of $S_2$ and original $S_1$, satisfies both sets of the original constraints; (b) the dotted domain $S_2'$ satisfies both sets of the original constraints in the two spaces. The original $S_2$, $A'B'C'D'E'$, includes an invalid subdomain $B'C'D'$ where the constraint $D_L \geq 0$ is violated.
Table 3. Randomly generated data for example 2

<table>
<thead>
<tr>
<th>Job number i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proc. time, ( p_i )</td>
<td>92</td>
<td>41</td>
<td>10</td>
<td>21</td>
<td>37</td>
<td>86</td>
<td>85</td>
<td>66</td>
<td>25</td>
<td>37</td>
</tr>
<tr>
<td>Due date, ( d_i )</td>
<td>90</td>
<td>95</td>
<td>116</td>
<td>64</td>
<td>151</td>
<td>171</td>
<td>66</td>
<td>97</td>
<td>49</td>
<td>93</td>
</tr>
<tr>
<td>Is ( d_i &lt; p_i )?</td>
<td>y</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this example, there are two Type-2 violations. It can be shown that the probability of a Type-2 violation is 0.184 for this particular parameter value combination; that is, on the average 1.84 jobs have \( d_i < p_i \). We also note that the values of \( \tau = 0.8 \) and \( \delta = 0.3 \) place this example in the domain of \( d_i > 0 \) (see Fig. 2), so that we do not have to be concerned with a Type-1 violation.

Now we consider feasible orderings of the four data parameters, \( p_L, p_U, d_L, \) and \( d_U \). Since \( p_L \leq p_U \) and \( d_L \leq d_U \), we are left with six possible permutations of the four parameters. Furthermore, we can assume that \( p_L \geq d_L \), and \( p_U \leq d_U \).

If \( p_L > d_L \), then the shortest due date will be smaller than the shortest processing time, which is unrealistic. Similarly, \( p_U > d_U \) is unrealistic. These two additional conditions reduce the feasible orderings to the following two:

\[
\begin{align*}
 p_L \leq p_U \leq d_L \leq d_U, \\
p_L \leq d_L \leq p_U \leq d_U.
\end{align*}
\]

(7) (8)

6.1. The case where \( p_L \leq p_U \leq d_L \leq d_U \)

In this case, \( d_i \geq p_i \) for every \( i \); hence, Type-2 violations never occur. The condition for which Relation (7) holds in terms of \( \tau \) and \( \delta \) can be determined as follows:

\[
p_U \leq d_L = n(p_L + p_U) \frac{2(1 - \tau) - \delta}{4}.
\]

(9)

Hence, we have that:

\[
2(1 - \tau) - \delta \geq \frac{4p_U}{n(p_L + p_U)}
\]

(a condition for which a Type-2 violation never occurs).

(10)

Several observations follow.

1. In practice, we often have some idea of what values should be assigned to \( n, \ p_L, \) and \( p_U \). Then it is easy to identify what domain in the \( (\tau, \delta) \) space satisfies Relation (10).

2. If \( p_L < p_U \) holds in Relation (10), then the right-hand side \( \approx 4/n \). If \( n \gg 1 \), then Relation (10) holds for most values of \( \tau \) and \( \delta \), assuming that the left-hand side > 0 and is not very small. If the left-hand side < 0, then both Type-1 and Type-2 violations can occur.

3. The conditions described here (such as Relations (7) and (10)) may or may not be realistic. For example, if \( p_L = 1 \) and \( p_U = 100 \), then selecting every \( d_i \geq 100 \) days even if some jobs require only one or two days of processing time, may not be realistic. Such anomalies are caused by the assumptions of Relation (7), particularly \( p_L \leq d_L \). To avoid these anomalies, we use Relation (8) as discussed in Section 6.2.

6.2. The case where \( p_L \leq d_L \leq p_U \leq d_U \)

We see that Type-2 violations can occur in this case because the relation \( d_i \leq p_U \) can cause \( d_i < p_i \) for some jobs, as observed in example 2. These violations are not caused by wrong combinations of values for \( \tau, \delta \); instead they are due to Relation (8). So, what can we do when Type-2 violations are found? First, this is an issue of finding algorithms to correct violations. The major focus of this article is how to detect possible violations by comparing the sample and parameter spaces. Second, correction algorithms depend on several factors such as the simplicity of the algorithms, computational efficiency, and probably most importantly, what kinds of models we want to simulate. Here we list a few simple remedies for a Type-2 violation.

As a simple brute-force approach, we may simply equate \( d_i \) and/or \( p_i \) whenever \( d_i < p_i \) occurs, either by \( d_i \leftarrow p_i \), or \( p_i \leftarrow d_i \), or \( d_i \leftarrow (d_i + p_i)/2 \). However, these corrections distort the originally intended properties of the data, such as the underlying distribution, the average due dates, etc. Another simple remedy that does not distort these properties would be to swap some violation samples with others. In example 2, we can swap \( d_i \) for \( i = 1 \) and 2, keeping \( p_i \) as in the original. Or, a Type-2 violation can occur since \( p_i \) and \( d_i \) are independently generated. We may generate \( p_i \) first, and then \( d_i \) may be generated by adding a non-negative random number to \( p_i \); so that \( d_i < p_i \) never occurs. Of course, such data would have different properties than originally intended.

We now determine how the inequality signs in Relation (8) relate the domains of \( \tau \) and \( \delta \). From the first inequality and Equation (4) for \( d_L \), we have that:

\[
p_L \leq d_L = n(p_L + p_U) \frac{2(1 - \tau) - \delta}{4}.
\]

(11)

Similarly, from the last inequality and Relation (5) for \( d_U \), we have that:

\[
p_U \leq d_U = n(p_L + p_U) \frac{2(1 - \tau) + \delta}{4}.
\]

(12)

From the middle inequality in Relation (8) and the inequality relations described in Relations (11) and (12), we can determine the conditions for \( \tau \) and \( \delta \) in terms of \( n, p_L, \) and \( p_U \):

\[
\frac{4p_U}{n(p_L + p_U)} \geq 2(1 - \tau) - \delta \geq \frac{4p_L}{n(p_L + p_U)},
\]

(13)

\[
2(1 - \tau) + \delta \geq \frac{4p_U}{n(p_L + p_U)}.
\]

(14)
Given \( n, p_U, \) and \( p_L, \) \( \tau \) and \( \delta \) must satisfy the two relations in Relation (13) and the relation in Relation (14). When \( n \gg 1 \) and \( p_U \gg p_L, \) the right-most side of Relation (13) \( \approx 0 \) (already imposed by Relation (4) to avoid Type-1 violations). Similarly, Relation (14) may also be satisfied by many \( (\tau, \delta) \) points. By adding the last two relations, we also have that:

\[
\tau \leq 1 - \frac{1}{n}.
\]  

(15)

We note that Relation (15) is not a sufficient condition for \( p_L \leq d_L \leq p_U \leq d_U; \) the set of relations in Relations (13) and (14) is necessary and sufficient.

For the uniform distributions of \( p_i \) and \( d_i, \) the probability of a Type-2 violation \((d_i < p_i)\) when \( p_L \leq d_L \leq p_U \leq d_U \) can be given as follows:

\[
P(d_i < p_i) = \frac{(p_U - d_i)^3}{2(p_U - p_i)(d_U - d_i)(p_U - d_i + 1)},
\]  

(16)

\[
\approx \frac{(p_U - d_i)^2}{2(p_U - p_i)(d_U - d_i)} \quad \text{(when } p_U - d_i \gg 1).\]  

(17)

The expected number of jobs for which \( d_i < p_i \) for a problem of \( n \) jobs is simply \( nP(d_i < p_i). \) When we use Equation (16) for example 2, we have \( P(d_i < p_i) = 0.184, \) as cited earlier.

7. Conclusions

We have discussed the concept of constraint isomorphism and showed how it can be applied to verify consistency among domains defined in different sample and parameter spaces. We cited some previous work for the total-tardiness problem that overlooked consistency violations. Detecting these types of errors is not necessarily trivial, especially when we work on characteristic parameters that are abstract and indirectly related to the original random data. Isomorphism tests help identify these errors for a wide spectrum of modeling and simulation problems involving random-data generation with constraints on different parameter sets.

Acknowledgements

The authors are grateful to Professor David Goldman, the anonymous referees and Professor Allan Warren of Cleveland State University for their valuable comments.

References


Biographies

Toshinori Munakata is a Professor in the Computer and Information Science Department at Cleveland State University, Cleveland, OH. His areas of interest include artificial intelligence, new computing paradigms...
and analysis of algorithms. He received his B.S., M.Sc., and Ph.D. from the University of Tokyo. He has served as the Guest Editor of three well-received Special Issues for the *Communications of the ACM*, focusing on real-world applications of artificial intelligence. He has also served as an Associate Editor for the *IEEE Expert* (currently *IEEE Intelligent Systems*). In addition to technical papers, he has published two books, most recently *Fundamentals of the New Artificial Intelligence: Beyond Traditional Paradigms*, Springer-Verlag, 1998. URL: http://cis.csuohio.edu/~munakata/

Adam Fadhila is a Professor of Computer and Information Science at Cleveland State University, Cleveland, OH. He holds an MBA in Finance and Decision Sciences from Miami University of Ohio, an M.Sc. in Computer Science and a Ph.D. in Computer Information Systems from the University of Cincinnati, OH. His current research interest is in the areas of decision support systems, artificial intelligence applications, knowledge discovery in databases, and medical information systems. His research has been published in many journals, including *Information and Management, Computers and Operations Research, Omega, Interfaces, Journal of the American Medical Informatics Association*, and *Information Systems Management*.

*Contributed by the Engineering Statistics and Applied Probability Department*