Revista de Matemática: Teoría y Aplicaciones 2010 **17**(1) : 53–68
cimpa – ucr issn: 1409-2433

CIRCULAR CHAINS OF CHINESE DICE

CADENAS CIRCULARES DE DADOS CHINOS

Eduardo Piza* Leo Schubert**

Received: 6 Jan 2009; Revised: 28 Oct 2009; Accepted: 30 Oct 2009

Keywords: Chinese dice, mixed-integer programming, simulated annealing, combinatorial optimization.

Palabras clave: Dados chinos, programación entera mixta, sobrecalentamiento simulado, recocido simulado, optimización combinatoria.

Mathematics Subject Classification: 05B99, 90C11, 90C27.

*CIMPA, Universidad de Costa Rica, 2060 San José, Costa Rica. Email: eduardo.piza@ucr.ac.cr

**University of Applied Sciences of Konstanz, Postfach 100543, D-78405 Konstanz, Germany. Email: schubert@htwg-konstanz.de

Abstract

In this paper we study *Chinese dice*, mathematical objects similar to ordinary dice but allowing repetition among their face values. We say that a die A is *preferred over* a die B (written $A \succ B$) if A wins more frequently than B does. We study first the existence of circular chains of three dice A, B, C (those that $A \succ B \succ C \succ A$) using a mixed integer programming algorithm. Then we generalize the problem to n-dimensional dice—that is, dice of n faces, with $n \geq 4$—and we search circular chains of length m (with $m \geq 3$) using a simulated annealing algorithm. We compare some different objective functions and obtain good solutions to the problem with very efficient algorithms. Finally we obtain a theoretical result concerning the existence of circular chains in the general case.

Resumen

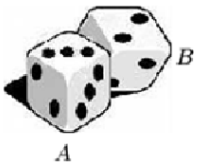
En este artículo estudiamos los *dados chinos*, objetos matemáticos similares a los dados ordinarios pero con la diferencia que pueden repetir algunos de sus lados. Decimos que el dado A es *preferido sobre* el dado B si A gana con mayor frecuencia que B. Estudiamos primero la existencia de cadenas circulares de tres dados A, B, C (aquellos para los cuales $A \succ B \succ C \succ A$) utilizando un algoritmo de programación lineal entera. Luego generalizamos el problema al caso de dados n-dimensionales, esto es, dados de n caras (con $n \geq 4$) y cadenas circulares de m dados (con $m \geq 3$), utilizando un algoritmo de recocido simulado. Comparamos diversas funciones objetivas y obtenemos buenas soluciones al problema con algoritmos eficientes. Finalmente obtenemos un resultado teórico acerca de la existencia de cadenas circulares para el caso general.

1 Introduction

Normal dice have the numbers $1, 2, 3, 4, 5, 6$ printed on their faces, without repetition, each face having the same probability output, and with opposite faces summing to 7. The *Chinese dice* also have numbers from $\{1, 2, 3, 4, 5, 6\}$ printed on their faces, but allow repetitions. We can represent Chinese dice as vectors in $\{1, 2, 3, 4, 5, 6\}^6$, each coordinate corresponding to a face. For instance, two dice A and B would be $(2, 3, 3, 2, 3, 6)$ and $(2, 2, 5, 5, 6, 1)$, as shown in Figure 1.

We say that a Chinese die A *is preferred* over Chinese die B (written $A \succ B$) if after rolling the dice, A wins more frequently than B, that is, the probability that A wins or ties is greater than $\frac{1}{2}$. For example, with the two Chinese dice $A = (2, 3, 3, 2, 3, 6)$ and B=$(2, 2, 5, 5, 6, 1)$ of

Figure 1, we can compute all the possible outputs (we assume that the dice outcomes are equally likely) in a matrix, as illustrated in Figure 1.



		Die A				
	2	**3**	**3**	**2**	**3**	**6**
2	0	1	1	0	1	1
2	0	1	1	0	1	1
5	−1	−1	−1	−1	−1	1
5	−1	−1	−1	−1	−1	1
6	−1	−1	−1	−1	−1	0
1	1	1	1	1	1	1

Die B

Figure 1: Comparing the Chinese dice $A = (2, 3, 3, 2, 3, 6)$ and B=$(2, 2, 5, 5, 6, 1)$. The values in the table have the following meaning: '1' means 'A wins', '0' means 'is a tie', '-1' means 'B wins'. In this example, A wins 16 times and loses only 15 times, so die A is preferred over die B: $A \succ B$.

Normal dice have the same strength among themselves: when comparing two "fair dice" we obtain 15 wins for each die and 6 ties. It is completely different when dealing with Chinese dice, where we can have extreme cases, for example $A = (6, 6, 6, 6, 6, 6)$ and $B = (1, 1, 1, 1, 1, 1)$, in which A wins against B all the time.

In this article we study the existence of *circular chains* of Chinese dice, for example three Chinese dice A, B, C such that $A \succ B \succ C \succ A$. The existence of these chains is somewhat surprising and seems to be a paradox, but it is not. We are interested in finding circular chains of Chinese dice in such a way that the differences in preference between dice will be as high as possible, but balanced among themselves. Later on, we formulate a generalization of these circular chains, considering dice with $n \geq 4$ faces and chains of length $m \geq 3$.

2 Solutions using mixed-integer programming

Let A, B, C be three Chinese dice. Our first two models for finding circular chains use the mixed-integer programming technique to maximize an objective function.

2.1 First model: maximizing the winning frequency

In this first approach we maximize the following objective function:

$$\max\ Z_{AB} + Z_{BC} + Z_{CA} - \lambda \cdot (v_1 + v_2 + v_3 + c_1 + c_2 + c_3), \qquad (1)$$

subject to the following restrictions:

$$X_{Ai} - X_{Bj} \leq (1 - \delta_{AiBj}) \cdot M, \quad \text{for } i, j = 1, \ldots, 6, \qquad (2a)$$
$$X_{Ai} - X_{Bj} \geq \epsilon - \delta_{AiBj} \cdot M, \quad \text{for } i, j = 1, \ldots, 6, \qquad (2b)$$
$$Z_{AB} + \textstyle\sum_{i,j}\ \delta_{AiBj} = 36. \qquad (3)$$

The corresponding restrictions for dice B-C and C-A are analogous to the restrictions in (2a), (2b) and (3). We also have 3 additional restrictions:

$$(Z_{AB} - Z_{BC}) + v_1 - c_1 = 0, \qquad (4a)$$
$$(Z_{AB} - Z_{CA}) + v_2 - c_2 = 0, \qquad (4b)$$
$$(Z_{BC} - Z_{CA}) + v_3 - c_3 = 0, \qquad (4c)$$

so, in total we have 222 different restrictions (73 restrictions in (2a), (2b) and (3) concerning the dice A-B and the same quantity for the dice B-C and C-A, plus the 3 restrictions in (4a), (4b), (4c)). The meaning of the variables and symbols is:

Z_{AB}: is the number of cases (from the 36 total cases) where die A wins against die B (analogous with the other quantities Z_{BC} and Z_{CA}). So, $Z_{AB}/36$ is the probability that A wins against B.

X_{Ai}: output value of the face i in die A. Similarly for X_{Bj}.

v_k, c_k: Slack-variables used to avoid a *void solution* (solution when a die wins in all 36 cases and other dice do not win in any case), $k = 1, 2, 3$.

λ: parameter used to avoid a void solution (for example, $\lambda = 100$).

δ_{AiBj}: 0-1 variable, which has a value 1 in the restrictions (2a) and (2b) when the die A with the face i cannot win against die B with the face j (similarly with pairs of dice B-C and C-A).

M: a "big" number (for example, $M = 6$).

ϵ: a "small" positive number (for example, $\epsilon = 0.5$).

We find the solution (see Figure 2) to this problem in approximately 15 minutes, using CPLEX, well-known specialized software for mixed-integer programming problems [2, 3, 4, 7], using an "average" home computer[1]. The solution found is a *global* optimal, as we can see by other methods. However, the computer process to formally verify that it is a global optimal solution was interrupted after more than 2 hours.

$$\begin{array}{llll} \text{Die } A = (2,2,3,3,6,6), & Z_{AB} = 20, & Z_{BA} = 16, & D_A = 4, \\ \text{Die } B = (1,1,5,5,5,5), & Z_{BC} = 20, & Z_{CB} = 16, & D_B = 4, \\ \text{Die } C = (3,3,4,4,4,6), & Z_{CA} = 20, & Z_{AC} = 10, & D_C = 10. \end{array}$$

Figure 2: The "4-4-10" solution found with our first model.

In Figure 2 the quantity D_A is defined as the difference $Z_{AB} - Z_{BA}$. Similarly, $D_B = Z_{BC} - Z_{CB}$ and $D_C = Z_{CA} - Z_{AC}$. The solution found is not unique, of course. We refer to this solution as the "4-4-10" solution, according to the values of D_A, D_B, D_C.

2.2 Modifications to speed the algorithm

In order to reduce the computer time in the previously mentioned model, we tried a slight modification of some of the parameters. In spite of this we did not find a significant reduction, although some reduction was found when changing the parameters to $M = 5$ in (2), $\epsilon = 0.01$ in (2b) and $\lambda = 0.25$ in (1). In fact, the computer time was reduced by approximately 10% in finding the same "4-4-10" solution shown in Figure 2, but still the whole process had to be interrupted after more than 2 hours, without the confirmation that the solution is a *global* maximum.

However, we found a significant reduction in computer time when we considered the following 27 additional restrictions. The idea is to eliminate some combinations of dice outcomes that do not have enough potential to win:

$$X_{A6} - X_{B4} \geq 0, \quad X_{B6} - X_{C4} \geq 0, \quad X_{C6} - X_{A4} \geq 0, \tag{5a}$$
$$X_{A5} - X_{B3} \geq 0, \quad X_{B5} - X_{C3} \geq 0, \quad X_{C5} - X_{A3} \geq 0, \tag{5b}$$
$$X_{A4} - X_{B2} \geq 0, \quad X_{B4} - X_{C2} \geq 0, \quad X_{C4} - X_{A2} \geq 0, \tag{5c}$$
$$X_{A3} - X_{B1} \geq 0, \quad X_{B3} - X_{C1} \geq 0, \quad X_{C3} - X_{A1} \geq 0, \tag{5d}$$

and for all $i < j$:

$$X_{Ai} - X_{Aj} \leq 0, \quad X_{Bi} - X_{Bj} \leq 0, \quad X_{Ci} - X_{Cj} \leq 0. \tag{6}$$

[1] A DELL laptop with 512 MB RAM and 2 GHz Intel processor.

Proposition 1: *The quality of the solutions of the model presented in section 2.1 is the same when we consider the additional 27 restrictions (5a), (5b), (5c), (5d) and (6).*

Proof: The last 15 restrictions in (6) only establish an order in the components of the vector, but do not change or restrict the quality of the solutions in the model. On the other hand, taking the first part of the restrictions in (5a), (5b), (5c) and (5d) together, we have

$$(X_{A6} \geq X_{B4}) \wedge (X_{A5} \geq X_{B3}) \wedge (X_{A4} \geq X_{B2}) \wedge (X_{A3} \geq X_{B1}).$$

With this restriction the feasible area is reduced by

$$\begin{aligned} &\neg[(X_{A6} \geq X_{B4}) \wedge (X_{A5} \geq X_{B3}) \wedge (X_{A4} \geq X_{B2}) \wedge (X_{A3} \geq X_{B1})] \\ &= \neg(X_{A6} \geq X_{B4}) \vee \neg(X_{A5} \geq X_{B3}) \vee \neg(X_{A4} \geq X_{B2}) \vee \neg(X_{A3} \geq X_{B1}) \\ &= (X_{A6} < X_{B4}) \vee (X_{A5} < X_{B3}) \vee (X_{A4} < X_{B2}) \vee (X_{A3} < X_{B1}). \end{aligned} \tag{7}$$

Then, with the inequalities in (6), the condition (7) implies:

$$\{X_{A1}, X_{A2}, X_{A3}, X_{A4}, X_{A5}, X_{A6}\} < \{X_{B4}, X_{B5}, X_{B6}\} \tag{8a}$$

$$\vee\{X_{A1}, X_{A2}, X_{A3}, X_{A4}, X_{A5}\} < \{X_{B3}, X_{B4}, X_{B5}, X_{B6}\} \tag{8b}$$

$$\vee\{X_{A1}, X_{A2}, X_{A3}, X_{A4}\} < \{X_{B2}, X_{B3}, X_{B4}, X_{B5}, X_{B6}\}. \tag{8c}$$

$$\vee\{X_{A1}, X_{A2}, X_{A3}\} < \{X_{B1}, X_{B2}, X_{B3}, X_{B4}, X_{B5}, X_{B6}\}. \tag{8d}$$

These last conditions (8a), (8b), (8c), (8d) show that in at least 18 cases (from the 36 total cases) the die A has to lose against the die B. In the condition (8a) three faces from die B have numbers greater than all the faces of die A. Then, A has to lose against B in at least 18 cases. Consequently, if die A never wins with the solutions that satisfy the restrictions in (5a)–(5d) and (6), then all the solutions with potential to win were in the feasible region. But also these restrictions eliminate the situations when die A cannot win or lose! The corresponding restrictions for the pairs of dice B-C and C-A yields analogous conditions and results. ■

With these additional restrictions, the computer time needed to find the optimal solution was between 1 and 6 seconds. In many cases 3 seconds were enough.

2.3 Second model: maximizing the winning differences

In order to find a circular of Chinese dice that maximizes the *winning differences* between dice (obtaining more "balanced" solutions), let us consider the following modification of the objective function in (1):

$$\max\ (1-\alpha)(Z_{AB}+Z_{BC}+Z_{CA})-\alpha(Z_{BA}+Z_{CB}+Z_{AC})-\lambda\cdot(v_1+v_2+v_3+c_1+c_2+c_3), \tag{9}$$

subject to the following restrictions:

$$X_{Ai} - X_{Bj} \leq (1 - \delta_{AiBj}) \cdot M, \quad \text{for } i,j = 1,\ldots,6, \tag{10a}$$

$$X_{Ai} - X_{Bj} \geq \epsilon - \delta_{AiBj} \cdot M, \quad \text{for } i,j = 1,\ldots,6, \tag{10b}$$

$$Z_{AB} + \textstyle\sum_{i,j}\ \delta_{AiBj} = 36. \tag{11}$$

The corresponding restrictions for dice B-C and C-A are analogous to the restrictions in (10a), (10b) and (11). We also have 3 additional restrictions:

$$(Z_{AB} - Z_{BA}) - (Z_{BC} - Z_{CB}) + v_1 - c_1 = 0, \tag{12a}$$

$$(Z_{AB} - Z_{BA}) - (Z_{CA} - Z_{AC}) + v_2 - c_2 = 0, \tag{12b}$$

$$(Z_{BC} - Z_{CB}) - (Z_{CA} - Z_{AC}) + v_3 - c_3 = 0, \tag{12c}$$

for a total of 222 restrictions. The meaning of the variables and parameters are the same as in Section 2.1. The new parameter α (with $0 \leq \alpha \leq 1$) weighs the relative importance of the winning frequencies of one die against the losing frequencies of the other corresponding die. By combining different values of α and λ we obtain the solutions shown in Table 1. In this figure we show four different complete balanced "5-5-5" solutions: in some sense these solutions are optimal. Values of $\lambda < \frac{1}{2}$ leads to "unbalanced" solutions, in contrast to values of $\lambda \geq \frac{1}{2}$, that produce "balanced" solutions.

3 General chains of n-dimensional Chinese dice

Our Chinese dice have two interesting sources of generalization:

1. To find circular chains of m Chinese dice A_1, A_2, ..., A_m, with $m > 3$: $A_1 \succ A_2 \succ \ldots \succ A_m \succ A_1$.

2. To work with general Chinese dice of n faces ($n \geq 4$), that is, n-dimensional Chinese dice, instead of the regular 6-face dice. In this case, a die is represented by a vector $(x_1, x_2, \ldots, x_n)$, where each coordinate $x_i \in \{1, 2, \ldots, n\}$ (this allows repetitions).

In Figure 3 we illustrated some n-dimensional Chinese dice. In order to find circular chains in these more general context, we need another

	$\lambda = 0.0$	$\lambda = 0.1$	$\lambda = 0.5$	$\lambda = 1.0$	$\lambda = 10.0$
$\alpha = 0.0$	A: 333333 B: 222222 C: 111466 "36-0-0" sol 20 sec.	A: 225555 B: 444444 C: 333366 "12-12-4" sol 79 sec.	A: 114446 B: 333336 C: 222555 "5-6-6" sol 153 sec.	A: 133366 B: 222555 C: 144444 "6-6-5" sol 156 sec.	A: 113555 B: 144444 C: 222266 "4-4-4" sol 220 sec.
$\alpha = 0.4$	A: 555555 B: 222222 C: 111666 "36-0-0" sol 123 sec.	A: 444444 B: 222266 C: 115555 "12-4-12" sol 228 sec.	A: 333336 B: 222555 C: 114446 "6-6-5" sol 832 sec.	A: 122555 B: 144444 C: 123366 **"5-5-5" sol** 694 sec.	A: 122555 B: 144444 C: 123366 **"5-5-5" sol** 733 sec.
$\alpha = 0.5$	A: 333666 B: 555555 C: 444444 "0-36-0" sol 112 sec.	A: 333333 B: 222266 C: 115555 "12-4-12" sol 107 sec.	A: 333336 B: 122556 C: 114446 **"5-5-5" sol** 133 sec.	A: 124444 B: 233336 C: 112555 **"5-5-5" sol** 144 sec.	A: 122555 B: 144444 C: 123366 "3-3-3" sol 123 sec.
$\alpha = 0.6$	A: 111666 B: 333333 C: 222222 "0-36-0" sol 510 sec.	A: 114555 B: 333333 C: 222266 "12-12-4" sol 349 sec.	A: 333466 B: 124555 C: 444444 "5-6-6" sol 450 sec.	A: 444455 B: 233556 C: 115555 "4-4-4" sol 500 sec.	A: 333345 B: 222555 C: 124446 "4-4-4" sol 528 sec.

Table 1: The principal solutions found with the model of Section 2.3.

kind of algorithm different from mixed-integer programming, because we experienced a fast increase in restrictions and computer time when m or n (or both) grew. For this purpose we employed a *simulated annealing* algorithm with great success, described in some detail in Section 4

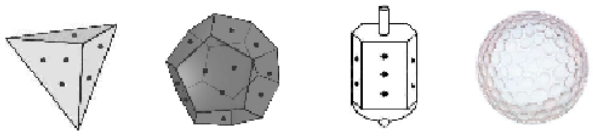


Figure 3: Some n-dimensional Chinese dice.

Given m Chinese dice of n-dimensions $A_1, A_2, \ldots, A_m$, let us define the quantities Z_i and $\bar{Z}_i$ (for $i = 1, \ldots, m$) in analogous form as in Section 2.1:

Z_i: is the number of cases (from the n^2 total cases) where Chinese die A_i wins against Chinese die A_{i+1} (or Chinese die A_1 when $i = m$).

Thus, the probability that A_i wins is Z_i/n^2.

$\bar{Z}_i$: is the number of cases where Chinese die A_{i+1} (or Chinese die A_1 when $i = m$) wins against Chinese die A_i.

Our primary objective is to find Chinese dice $A_1, A_2, \ldots, A_m$ such that the quantities $Z_i - \bar{Z}_i$ are all positive. Furthermore, we also want to find *optimal* solutions, in the sense that these differences $Z_i - \bar{Z}_i$ are *maximal* and *balanced* (approximately equal among themselves). To achieve these aims, we study the maximization of two different objective functions, using a simulated annealing algorithm described in Section 4.

3.1 Maximizing the sum of the differences $Z_i - \bar{Z}_i$

We consider the semi-concave function $f(x) = \sqrt{x}$, if $x > 0$, and $f(x) = x$, if $x \leq 0$, as shown in Figure 4. In this approach our problem is to maximize the objective function

$$f(Z_1 - \bar{Z}_1) + f(Z_2 - \bar{Z}_2) + \cdots + f(Z_m - \bar{Z}_m), \tag{13}$$

over all the possible n-dimensional Chinese dice $A_1, A_2, \ldots, A_m$. The selection of this specific function f fits the desire to penalize negative values of the difference $Z_i - \bar{Z}_i$ and at the same time to favour the approximate equality among these quantities. We tested some other similar functions instead of f, for example $\hat{f}(x) = \ln(1+x)$, if $x > 0$ and $\hat{f}(x) = x$, if $x \leq 0$. In the end we obtained equivalent solutions with all these kinds of semi-concave functions, and we decided to select the function f described above due to its simplicity, from the point of view of computer time needed to evaluated it.

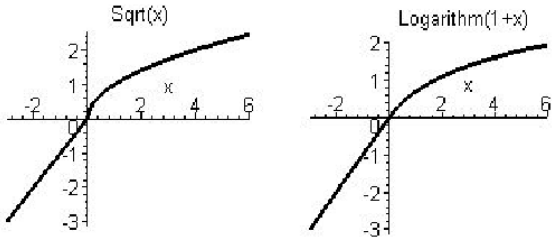


Figure 4: The semi-concave functions f and $\hat{f}$ of our first approach to find circular chains of n-dimensional Chinese dice $A_1, A_2, \ldots, A_m$.

Using this approach we find solutions very quickly, some of them are shown in Table 2.

$n = 4$ dimensions		$n = 5$ dimensions		$n = 6$ dimensions	
$m = 3$	$m = 4$	$m = 3$	$m = 4$	$m = 3$	$m = 4$
A_1: 1144	A_1: 3232	A_1: 44411	A_1: 25225	A_1: 622622	A_1: 333333
A_2: 3313	A_2: 2242	A_2: 33333	A_2: 11555	A_2: 115555	A_2: 226262
A_3: 2242	A_3: 4141	A_3: 22525	A_3: 44444	A_3: 343334	A_3: 551515
	A_4: 1333		A_4: 33333		A_4: 444444
"2-2-2"	"2-2-2-2"	"5-5-7"	"1-5-25-5"	"4-12-12"	"12-4-12-36"
0.00 sec.	0.27 sec.	11.65 sec.	23.78 sec.	16.59 sec.	52.78 sec.

Table 2: Some solutions obtained by our simulated annealing algorithm, using the semi-concave functions f. For example, for $n = 6$ and $n = 3$ we obtain a "4-12-12" solution: the numbers inside quotes denote the quantities $Z_i - \bar{Z}_i$.

3.2 Finding more balanced solutions

The solutions obtained by the preceding model are not as balanced as we would like. In order to find more balanced solutions, we changed the objective function (to be maximized) as follows:

$$\sum_{i=1}^{m} \lambda \cdot (Z_i - \bar{Z}_i) - (1 - \lambda) \cdot |(Z_i - \bar{Z}_i) - (Z_{i+1} - \bar{Z}_{i+1})|, \qquad (14)$$

where $\lambda \in (0, 1)$ is a fixed parameter. In this way we penalize unbalanced solutions. Our simulated annealing algorithm quickly produced very balanced solutions using this objective function, as shown in Table 3.

$n = 4$ dimensions		$n = 5$ dimensions		$n = 6$ dimensions	
$m = 3$	$m = 4$	$m = 3$	$m = 4$	$m = 3$	$m = 4$
B_1: 1441	C_1: 4141	A_1: 33333	A_1: 33333	A_1: 363333	A_1: 334332
B_2: 3313	C_2: 3331	A_2: 52225	A_2: 33522	A_2: 125562	A_2: 626222
B_3: 2224	C_3: 3232	A_3: 44115	A_3: 11255	A_3: 141644	A_3: 515512
	C_4: 2242		A_4: 14144		A_4: 414144
"2-2-2"	"2-2-2-2"	"5-5-5"	"5-5-3-5"	"5-5-5"	"8-8-8-8"
0.33 sec.	2.25 sec.	10.49 sec.	15.38 sec.	17.74 sec.	45.92 sec.

Table 3: Modification of the objective function in (14) produces more balanced solutions. Here we show some results only for $\lambda = 0.5$.

3.3 Theoretical results

Our simulated annealing algorithm can compute good and balanced circular chains of Chinese dice for a large variety of values of $n \geq 4$ and $m \geq 3$, in a relative low time. On the other hand, we have the following theoretical result, that guarantees the existence of circular chains of arbitrary length, for almost all dimensions.

Proposition 2: *For all $n \geq 4$ and $m \geq 3$ there exist circular chains of n-dimensional Chinese dice of length m: $A_1 \succ A_2 \succ \cdots \succ A_m \succ A_1$, with the only exception being $(n, m) = (4, 5)$, a case which does not admit a solution at all.*

Proof: Let $m \geq 3$ be an integer. First notice that if we find m Chinese dice $D_1, D_2, \ldots, D_m$ of 4 dimensions (that is, $n = 4$) that form a circular chain,

$$D_1 \succ D_2 \succ \cdots \succ D_m \succ D_1,$$

then we can fill the rest of the coordinates with the number 1 (or any other fixed number) to form a circular chain of Chinese dice of n dimensions:

$$(D_1, 1, \ldots, 1) \succ (D_2, 1, \ldots, 1) \succ \cdots \succ (D_m, 1, \ldots, 1) \succ (D_1, 1, \ldots, 1).$$

Then, it is sufficient to work with $n = 4$ dimensions. Second, notice that if $B_1 \succ B_2 \succ B_3 \succ B_1$ is a circular chain of length 3, then we can use it repetitively p times,

$$B_1 \succ B_2 \succ B_3 \, [\succ B_1 \succ B_2 \succ B_3]^{p-1},$$

to form a circular chain of length $3p$, for any positive integer p. Analogously, if $C_1 \succ C_2 \succ C_3 \succ C_4 \succ C_1$ is a circular chain of length 4, we can use it repetitively q times,

$$C_1 \succ C_2 \succ C_3 \succ C_4 \, [\succ C_1 \succ C_2 \succ C_3 \succ C_4]^{q-1},$$

to form a circular chain of length $4q$, for any positive integer q. If we are able to "combine" these circular chains in the extremes, then we can construct circular chains of any arbitrary length of the form $3p + 4q$. Precisely this is the case with the particular circular chains of Chinese dice B_1, B_2, B_3 and C_1, C_2, C_3, C_4 presented in Table 3, because they verify that $B_3 \succ C_1$ by 2 units and $C_4 \succ B_1$ also by 2 units, as we can easily check. We illustrated this in Figure 5. So, all the chains produced with these particular generators are circular and completely balanced.

Third, every integer $m \geq 3$ if it is not equal to 5 then it can be written in the form $m = 3p + 4q$, for some positive integers p and q. We prove this last assertion by induction on $m \geq 3$, as follows: (*i*) *Initial steps*: for $m = 3$ or $m = 4$ the affirmation is clear, because $3 = 3 \cdot 1 + 4 \cdot 0$ and $4 = 3 \cdot 0 + 4 \cdot 1$; (*ii*) *Induction hypothesis*: suppose the integer $m \geq 6$ has the form $m = 3p + 4q$, where p and q are natural numbers, not both of them null; (*iii*) *Inductive step*: by the induction hypothesis, some of the numbers p or q are positive. Moreover, note that if $p = 0$ then $q \geq 2$. Thus, we have:

$$m + 1 = 3p + 4q + 1 = \begin{cases} 3(p-1) + 4(q+1) & \text{if } p > 0, \\ 3 \cdot 3 + 4(q-2) & \text{if } p = 0 \text{ and } q \geq 2. \end{cases}$$

That proves the assertion. Fourth, we can find the existence of circular chains of 5 Chinese dice in 5 or more dimensions with the aim of our simulated annealing algorithm. In fact, a good semi-balanced "5-5-3-4-4" particular solution is: $D_1 = (3, 1, 4, 3, 4)$, $D_2 = (3, 3, 3, 3, 3)$, $D_3 = (2, 2, 2, 5, 5)$, $D_4 = (5, 1, 5, 4, 1)$, $D_5 = (4, 1, 4, 4, 3)$. Finally, the only case for which there does not exist a solution at all occurs when $(n, m) = (4, 5)$. This can be checked by enumeration, using for example the CPLEX software. ■

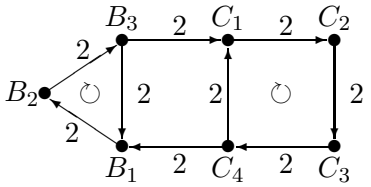


Figure 5: The Chinese dice B_1, B_2, B_3 and C_1, C_2, C_3, C_4 shown in Table 3 have a perfect match in the extremes that allows producing other circular chains of arbitrary length. All the dice are preferred to the following die by exactly 2 units, producing totally balanced circular chains.

4 Simulated annealing algorithm

We successfully implemented a simulated annealing algorithm to work with circular chains of n-dimensional Chinese dice of length m.

Simulated annealing is a generic probabilistic meta-heuristic algorithm for the global optimization problem [1, 6], namely locating a good approx-

imation to the global optimum of a given function in a large search space. It was independently presented by Kirkpatrick, Gelatt and Vecchi in 1983, and by Černý in 1985. The method is an adaptation of the Metropolis-Hastings algorithm, a Monte Carlo method to generate sample states of a thermodynamic system, invented by Metropolis et al. in 1953. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat causes the atoms to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly through states of higher energy; the slow cooling gives them more chances of finding configurations with lower internal energy than the initial one.

By analogy with this physical process, each step of the simulated annealing algorithm replaces the current solution by a random "nearby" solution, chosen with a probability that depends on the difference between the corresponding function values and on a global parameter T (called the "temperature"), that is gradually decreased during the process. The dependency is such that the current solution changes almost randomly when T is large, but increasingly "downhill" as T goes to zero. The allowance for "uphill" moves saves the method from becoming stuck at local minima—which are the bane of greedier methods.

The algorithm starts selecting at random a collection of m Chinese dice of n dimensions. A series of iterations begins, modifying the Chinese dice according to the following rules:

1. Select at random a Chinese die i, with $i \in \{1, \ldots, m\}$.
2. Select at random the face (or position) j of the i-th Chinese die, with $j \in \{1, \ldots, n\}$.
3. Select at random a new value $k' \in \{1, \ldots, n\}$ that modifies the actual value k in the j-th position of the i-th Chinese die.
4. We accept this proposed change with probability $\min\{1, e^{\Delta/T}\}$, where Δ is the change in the value of the objective function (see (13) and (14)), and $T > 0$ is the temperature parameter.

In this way, if the newly generated Chinese die *increases* the actual value of the objective function, then it is always accepted (with probability 1). On the other hand, if the newly generated Chinese die *decreases* the value of the objective function, it may still be accepted, but only with

probability $e^{\Delta/T}$ (in this case $\Delta < 0$). This acceptance criteria is called the *Metropolis rule*. Note that when the value of the temperature T is close to 0 (but positive), then the probability of accepting bad changes is close to 0.

It is well known (see [1, 6]) that in theory this "random walk" converges asymptotically to a global maximum of the objective function with probability 1, when certain conditions are satisfied. The first general condition is that the temperature $T > 0$ has to tend to 0 "slowly". A homogeneous Markov chain is associated with each value of the temperature T and the second general condition for convergence to a global maximum is that all these Markov chains have stationary distributions. That is the case with the algorithm we implemented, because we have total connectivity and reversibility among the dice configurations and also the probabilities of reaching any "neighboring" Chinese die are all equal. In practice we can destroy the convergence of the algorithm, when we implement convergence of the temperature parameter T to 0 or when we simulate the convergence process of each Markov chain to their stationary distribution.

In our simulated annealing algorithm we use the following "cooling scheme":

Initial temperature: the initial temperature T_0 is selected at the beginning in order to let the Metropolis rule be more tolerant, accepting nearly $\chi \times 100\%$ of new "bad" Chinese dice. We run a sequence of preliminary "phantom steps" in order to select χ with this prerequisite.

Decrease of temperature: every certain number of steps the system is cooled down a little, decreasing the value of the temperature T_k using the geometrical cooling scheme: $T_{k+1} = \lambda \cdot T_k$, where λ is a constant previously chosen empirically between $[0.92, 0.98]$ ($\lambda = 0.95$ was a good selection in all our experiments).

Length of the temperature chains: the temperature parameter is updated each nLIMIT steps, or when it has already accepted nOVER new "bad" changes. We successfully used empirical values of nLIMIT $\in [10^5, 10^6]$ and nOVER $\in [10000, 20000]$, depending on the size of the problem.

Criteria to stop the algorithm: a maximum of 150 cycles of temperature are completed, because in practice the quantity $T_{150} = (T_0)^{150}$ is almost null independently of the initial value T_0. Nevertheless, if

for the last NCAD temperature steps a new "good" Chinese die does not appear, then the process is stopped. We used NCAD = 3 in our experiments with good results.

The algorithm extensively used an additive random generation algorithm, recommended by Knuth [5]. The results of our algorithm are very satisfactory, reaching solutions as good as the ones obtained by mixed-integer programming, but in less computer time. Some of these solutions are already presented in Tables 2 and 3. In Figure 6 we present a solution found by our simulated annealing algorithm for non trivial length of n and m, as an example.

Die 1:	4, 4, 4, 4, 5, 4, 4, 8	Die 6:	4, 4, 4, 8, 4, 4, 4, 4	Die 11:	4, 4, 4, 4, 4, 4, 4, 8
Die 2:	3, 3, 8, 8, 3, 7, 3, 3	Die 7:	8, 8, 3, 3, 7, 3, 3, 3	Die 12:	3, 3, 3, 3, 3, 7, 8, 8
Die 3:	2, 7, 7, 7, 2, 2, 2, 7	Die 8:	7, 2, 7, 7, 7, 2, 2, 2	Die 13:	7, 2, 7, 2, 7, 7, 2, 2
Die 4:	6, 6, 1, 6, 1, 2, 6, 6	Die 9:	6, 1, 6, 6, 2, 1, 6, 6	Die 14:	6, 6, 1, 6, 1, 2, 6, 6
Die 5:	5, 5, 5, 3, 5, 5, 1, 5	Die 10:	3, 5, 5, 5, 5, 5, 1, 5	Die 15:	5, 4, 5, 5, 5, 1, 5, 5

Figure 6: Solution found in 249 seconds by our simulated annealing algorithm, for $n = 8$ and $m = 15$. In this example, all the differences $Z_i - \bar{Z}_i$ are equal to 20: it is a "20-20-. . .-20" solution.

5 Final remarks

The popular children's game "rock, paper, scissors" is an example of "normal" dice in 3 dimensions: each "die" (a hidden hand) has non-repetitive values and therefore has the same probability to win against the other. The non-transitivity of the "face" values ("rock" $\succ$ "scissors" $\succ$ "paper" $\succ$ "rock") is the only important mathematical concept involved. Circular chains do not exist for 3 dimensions.

In art, some famous paintings by Escher remind us of the Chinese dice. For example his famous "Ascending and Descending" painting (see Figure 7) has some resemblance to Chinese dice in 4 dimensions. However, the "random factor" inherent in the concept of Chinese dice is still missing.

We think there are some possible applications of the circular chains of Chinese dice inside the theory of multi-objective optimization and also in the utility theory in economics. We are hopeful to find some application in this context.

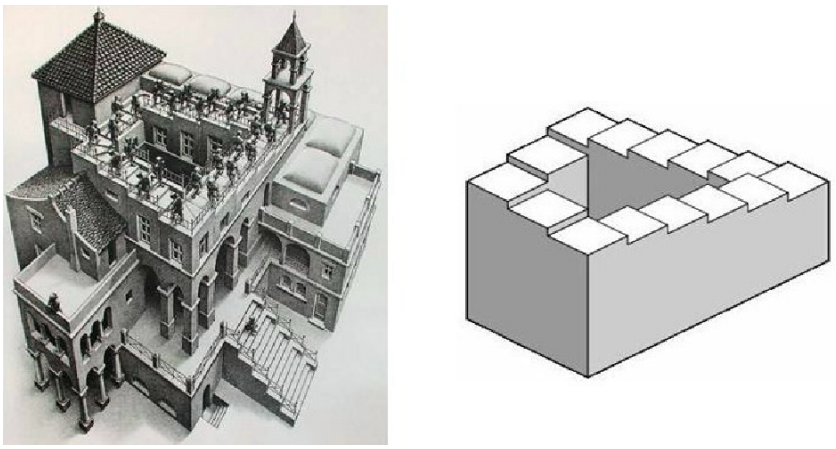


Figure 7: "Ascending and Descending" by Escher (1960) and the Penrose stairs optical illusion (1958).

References

[1] Aarts, E.; Korst, J. (1990) *Simulated Annealing and Boltzmann Machines. A Stochastic Approach to Combinatorial Optimization and Neural Computing.* John Wiley & Sons, Chichester.

[2] Bixby, R.E. (1992) "Implementing the simplex method: The initial basis", *ORSA Journal on Computing* **4**: 267–284.

[3] Bixby, R.E. et al. (2002) *ILOG AMPL CPLEX System, Version 8.0.* User's Guide.

[4] Fourer, R.; Gay, D.M.; Kernighan, B.W. (2002) *AMPL: A Modeling Language for Mathematical Programming.* Duxbury Press-Brooks-Cole Publishing Co.

[5] Knuth, D.E. (1981) *Seminumerical Algorithms. Second edition, volume 2 of the book The Art of Computer Programming.* Addison-Wesley, Reading, Mass.

[6] Laarhoven, P.J.M. van (1988) "Theoretical and computational aspects of simulated annealing", Centrum voor Wiskunde in Informatic, Tract 57.

[7] Williams, H.P. (1999) *Model Building in Mathematical Programming.* John Wiley & Sons, Chinchester.