Reconstruction of complete interval tournaments. II.

Antal Iványi
Eötvös Loránd University,
Department of Computer Algebra
1117 Budapest, Pázmány P. sétány 1/C.
email: tony@compalg.inf.elte.hu

Abstract. Let $a, b$ ($b \geq a$) and $n$ ($n \geq 2$) be nonnegative integers and let $T(a, b, n)$ be the set of such generalised tournaments, in which every pair of distinct players is connected at most with $b$, and at least with $a$ arcs. In [40] we gave a necessary and sufficient condition to decide whether a given sequence of nonnegative integers $D = (d_1, d_2, \ldots, d_n)$ can be realized as the out-degree sequence of a $T \in T(a, b, n)$. Extending the results of [40] we show that for any sequence of nonnegative integers $D$ there exist $f$ and $g$ such that some element $T \in T(g, f, n)$ has $D$ as its out-degree sequence, and for any $(a, b, n)$-tournament $T'$ with the same out-degree sequence $D$ hold $a \leq g$ and $b \geq f$. We propose a $\Theta(n)$ algorithm to determine $f$ and $g$ and an $O(d_n n^2)$ algorithm to construct a corresponding tournament $T$.

1 Introduction

Let $a, b$ ($b \geq a$) and $n$ ($n \geq 2$) be nonnegative integers and let $T(a, b, n)$ be the set of such generalised tournaments, in which every pair of distinct players is connected at most with $b$, and at least with $a$ arcs. The elements of $T(a, b, n)$ are called $(a, b, n)$-tournaments. The vector $D = (d_1, d_2, \ldots, d_n)$ of the out-degrees of $T \in T(a, b, n)$ is called the score vector of $T$. If the elements of $D$ are in nondecreasing order, then $D$ is called the score sequence of $T$.

2010 Mathematics Subject Classification: 05C20, 68C25
Key words and phrases: score sequences, tournaments, efficiency of algorithms

47
An arbitrary vector $D = (d_1, d_2, \ldots, d_n)$ of nonnegative integers is called \textit{graphical vector}, iff there exists a loopless multigraph whose degree vector is $D$, and $D$ is called \textit{digraphical vector} (or \textit{score vector}) iff there exists a loopless directed multigraph whose out-degree vector is $D$.

A nondecreasingly ordered graphical vector is called \textit{graphical sequence}, and a nondecreasingly ordered digraphical vector is called \textit{digraphical sequence} (or \textit{score sequence}).

The number of arcs of $T$ going from player $P_i$ to player $P_j$ is denoted by $m_{ij}$ ($1 \leq i, j \leq n$), and the matrix $M = [1 \cdots n, 1 \cdots n]$ is called \textit{point matrix} or \textit{tournament matrix} of $T$.

In the last sixty years many efforts were devoted to the study of both types of vectors, resp. sequences. E.g. in the papers [8, 16, 18, 19, 20, 21, 26, 30, 32, 34, 36, 45, 68, 84, 85, 88, 90, 98] the graphical sequences, while in the papers [1, 2, 3, 7, 8, 11, 17, 27, 28, 29, 31, 33, 37, 49, 48, 50, 55, 58, 57, 60, 61, 62, 64, 65, 66, 69, 78, 79, 82, 94, 86, 87, 97, 100, 101] the score sequences were discussed.

Even in the last two years many authors investigated the conditions, when $D$ is graphical (e.g. [4, 9, 12, 13, 22, 23, 24, 25, 38, 39, 43, 47, 51, 52, 59, 75, 81, 92, 93, 95, 96, 104]) or digraphical (e.g. [5, 35, 40, 46, 54, 56, 63, 67, 70, 71, 72, 73, 74, 83, 87, 89, 102]).

In this paper we deal only with directed graphs and usually follow the terminology used by K. B. Reid [79, 80]. If in the given context $a$, $b$ and $n$ are fixed or non important, then we speak simply on \textit{tournaments} instead of generalised or $(a, b, n)$-tournaments.

We consider the loopless directed multigraphs as generalised tournaments, in which the number of arcs from vertex/player $P_i$ to vertex/player $P_j$ is denoted by $m_{ij}$, where $m_{ij}$ means the number of points won by player $P_i$ in the match with player $P_j$.

The first question: how one can characterise the set of the score sequences of the $(a, b, n)$-tournaments. Or, with another words, for which sequences $D$ of nonnegative integers does exist an $(a, b, n)$-tournament whose out-degree sequence is $D$. The answer is given in Section 2.

If $T$ is an $(a, b, n)$-tournament with point matrix $M = [1 \cdots n, 1 \cdots n]$, then let $E(T)$, $F(T)$ and $G(T)$ be defined as follows: $E(T) = \max_{1 \leq i, j \leq n} m_{ij}$, $F(T) = \max_{1 \leq i < j \leq n} (m_{ij} + m_{ji})$, and $g(T) = \min_{1 \leq i < j \leq n} (m_{ij} + m_{ji})$. Let $\Delta(D)$ denote the set of all tournaments having $D$ as out-degree sequence, and let $e(D)$, $f(D)$ and $g(D)$ be defined as follows: $e(D) = \{\min E(T) \mid T \in \Delta(D)\}$, $f(D) = \{\min F(T) \mid T \in \Delta(D)\}$, and $g(D) = \{\max G(T) \mid T \in \Delta(D)\}$. In the sequel we use the short notations $E$, $F$, $G$, $e$, $f$, $g$, and $\Delta$. 
Hulett et al. [39, 99], Kapoor et al. [44], and Tripathi et al. [91, 92] investigated the construction problem of a minimal size graph having a prescribed degree set [77, 103]. In a similar way we follow a mini-max approach formulating the following questions: given a sequence $D$ of nonnegative integers,

- How to compute $e$ and how to construct a tournament $T \in \Delta$ characterised by $e$? In Section 3 a formula to compute $e$, and an algorithm to construct a corresponding tournament are presented.

- How to compute $f$ and $g$? In Section 4 an algorithm to compute $f$ and $g$ is described.

- How to construct a tournament $T \in \Delta$ characterised by $f$ and $g$? In Section 5 an algorithm to construct a corresponding tournament is presented and analysed.

We describe the proposed algorithms in words, by examples and by the pseudocode used in [14]. Researchers of these problems often mention different applications, e.g. in biology [55], chemistry Hakimi [32], and Kim et al. in networks [47].

2 Existence of a tournament with arbitrary degree sequence

Since the numbers of points $m_{ij}$ are not limited, it is easy to construct a $(0, d_n, n)$-tournament for any $D$.

**Lemma 1** If $n \geq 2$, then for any vector of nonnegative integers $D = (d_1, d_2, \ldots, d_n)$ there exists a loopless directed multigraph $T$ with out-degree vector $D$ so, that $E \leq d_n$.

**Proof.** Let $m_{n1} = d_n$ and $m_{i,i+1} = d_i$ for $i = 1, 2, \ldots, n - 1$, and let the remaining $m_{ij}$ values be equal to zero. ■

Using weighted graphs it would be easy to extend the definition of the $(a, b, n)$-tournaments to allow arbitrary real values of $a$, $b$, and $D$. The following algorithm *Naive-Construct* works without changes also for input consisting of real numbers.

We remark that Ore in 1956 [66] gave the necessary and sufficient conditions of the existence of a tournament with prescribed in-degree and out-degree vectors. Further Ford and Fulkerson [17, Theorem11.1] published in 1962
necessary and sufficient conditions of the existence of a tournament having prescribed lower and upper bounds for the in-degree and out-degree of the vertices. They results also can serve as basis of the existence of a tournament having arbitrary out-degree sequence.

2.1 Definition of a naive reconstructing algorithm

Sorting of the elements of $D$ is not necessary.

**Input.** $n$: the number of players ($n \geq 2$);
$D = (d_1, d_2, \ldots, d_n)$: arbitrary sequence of nonnegative integer numbers.

**Output.** $\mathcal{M} = [1..n, 1..n]$: the point matrix of the reconstructed tournament.

**Working variables.** $i, j$: cycle variables.

**Naive-Construct**($n, D$)

01 for $i \leftarrow 1$ to $n$
02     for $j \leftarrow 1$ to $n$
03         do $m_{ij} \leftarrow 0$
04     $m_{n1} \leftarrow d_n$
05 for $i \leftarrow 1$ to $n - 1$
06     do $m_{i,i+1} \leftarrow d_i$
07 return $\mathcal{M}$

The running time of this algorithm is $\Theta(n^2)$ in worst case (in best case too).

Since the point matrix $\mathcal{M}$ has $n^2$ elements, this algorithm is asymptotically optimal.

3 Computation of $e$

This is also an easy question. From here we suppose that $D$ is a nondecreasing sequence of nonnegative integers, that is $0 \leq d_1 \leq d_2 \leq \ldots \leq d_n$. Let $h = \lceil d_n/(n-1) \rceil$.

Since $\Delta(D)$ is a finite set for any finite score vector $D$, $e(D) = \min\{E(T) | T \in \Delta(D)\}$ exists.

**Lemma 2** If $n \geq 2$, then for any sequence $D = (d_1, d_2, \ldots, d_n)$ there exists a $(0, b, n)$-tournament $T$ such that

$$E \leq h \quad \text{and} \quad b \leq 2h,$$

(1)
and $h$ is the smallest upper bound for $e$, and $2h$ is the smallest possible upper bound for $b$.

**Proof.** If all players gather their points in a uniform as possible manner, that is
\[
\max_{1 \leq j \leq n} m_{ij} - \min_{1 \leq j \leq n, i \neq j} m_{ij} \leq 1 \quad \text{for } i = 1, 2, \ldots, n, \tag{2}
\]
then we get $E \leq h$, that is the bound is valid. Since player $P_n$ has to gather $d_n$ points, the pigeonhole principle \cite{6, 15, 42} implies $E \geq h$, that is the bound is not improvable. $E \leq h$ implies $\max_{1 \leq i < j \leq n} m_{ij} + m_{ji} \leq 2h$. The score sequence $D = (d_1, d_2, \ldots, d_n) = (2n(n - 1), 2n(n - 1), \ldots, 2n(n - 1))$ shows, that the upper bound $b \leq 2h$ is not improvable. ■

**Corollary 1** If $n \geq 2$, then for any sequence $D = (d_1, d_2, \ldots, d_n)$ holds $e(D) = \lceil d_n/(n - 1) \rceil$.

**Proof.** According to Lemma 2 $h = \lceil d_n/(n - 1) \rceil$ is the smallest upper bound for $e$. ■

### 3.1 Definition of a construction algorithm

The following algorithm constructs a $(0, 2h, n)$-tournament $T$ having $E \leq h$ for any $D$.

**Input.** $n$: the number of players ($n \geq 2$);
$D = (d_1, d_2, \ldots, d_n)$: arbitrary sequence of nonnegative integer numbers.

**Output.** $M = [1 \ldots n, 1 \ldots n]$: the point matrix of the tournament.

**Working variables.** $i$, $j$, $l$: cycle variables;
$k$: the number of the ”larger parts” in the uniform distribution of the points.

**Pigeonhole-Construct**($n$, $D$)

01 for $i \leftarrow 1$ to $n$
02 \hspace{1em} do $m_{ii} \leftarrow 0$
03 \hspace{1em} do $k \leftarrow d_i - (n - 1)\lfloor d_i/(n - 1) \rfloor$
04 \hspace{1em} for $j \leftarrow 1$ to $k$
05 \hspace{2em} do $l \leftarrow i + j \mod n$
06 \hspace{2em} do $m_{ll} \leftarrow \lfloor d_n/(n - 1) \rfloor$
07 \hspace{1em} for $j \leftarrow k + 1$ to $n - 1$
08 \hspace{2em} do $l \leftarrow i + j \mod n$
09 \hspace{2em} do $m_{ll} \leftarrow \lceil d_n/(n - 1) \rceil$
10 return $M$
The running time of PIGEONHOLE-CONSTRUCT is $\Theta(n^2)$ in worst case (in best case too). Since the point matrix $M$ has $n^2$ elements, this algorithm is asymptotically optimal.

4 Computation of $f$ and $g$

Let $S_i$ $(i = 1, 2, \ldots, n)$ be the sum of the first $i$ elements of $D$, $B_i$ $(i = 1, 2, \ldots, n)$ be the binomial coefficient $n(n-1)/2$. Then the players together can have $S_n$ points only if $fB_n \geq S_n$. Since the score of player $P_n$ is $d_n$, the pigeonhole principle implies $f \geq \lceil d_n/(n-1) \rceil$.

These observations result the following lower bound for $f$:

$$f \geq \max \left( \left\lceil \frac{S_n}{B_n} \right\rceil, \left\lceil \frac{d_n}{n-1} \right\rceil \right).$$

(3)

If every player gathers his points in a uniform as possible manner then

$$f \leq 2 \left\lceil \frac{d_n}{n-1} \right\rceil.$$

(4)

These observations imply a useful characterisation of $f$.

**Lemma 3** If $n \geq 2$, then for arbitrary sequence $D = (d_1, d_2, \ldots, d_n)$ there exists a $(g, f, n)$-tournament having $D$ as its out-degree sequence and the following bounds for $f$ and $g$:

$$\max \left( \left\lceil \frac{S_n}{B_n} \right\rceil, \left\lceil \frac{d_n}{n-1} \right\rceil \right) \leq f \leq 2 \left\lceil \frac{d_n}{n-1} \right\rceil,$$

(5)

$$0 \leq g \leq f.$$

(6)

**Proof.** (5) follows from (3) and (4), (6) follows from the definition of $f$. ■

It is worth to remark, that if $d_n/(n-1)$ is integer and the scores are identical, then the lower and upper bounds in (5) coincide and so Lemma 3 gives the exact value of $F$.

In connection with this lemma we consider three examples. If $d_i = d_n = 2c(n-1)$ ($c > 0$, $i = 1, 2, \ldots, n-1$), then $d_n/(n-1) = 2c$ and $S_n/B_n = c$, that is $S_n/B_n$ is twice larger than $d_n/(n-1)$. In the other extremal case, when $d_i = 0$ ($i = 1, 2, \ldots, n-1$) and $d_n = cn(n-1) > 0$, then $d_n/(n-1) = cn$, $S_n/B_n = 2c$, so $d_n/(n-1)$ is $n/2$ times larger, than $S_n/B_n$. 

Reconstruction of complete interval tournaments. II.

If \( D = (0, 0, 0, 40, 40, 40) \), then Lemma 3 gives the bounds \( 8 \leq f \leq 16 \). Elementary calculations show that Figure 1 contains the solution with minimal \( f \), where \( f = 10 \).

In [40] we proved the following assertion.

**Theorem 1** For \( n \geq 2 \) a nondecreasing sequence \( D = (d_1, d_2, \ldots, d_n) \) of nonnegative integers is the score sequence of some \((a, b, n)\)-tournament if and only if

\[
aB_k \leq \sum_{i=1}^{k} d_i \leq bB_n - L_k - (n - k)d_k \quad (1 \leq k \leq n),
\]

(7)

where

\[
L_0 = 0, \quad \text{and} \quad L_k = \max \left( L_{k-1}, \quad bB_k - \sum_{i=1}^{k} d_i \right) \quad (1 \leq k \leq n).
\]

(8)

The theorem proved by Moon [61], and later by Kemnitz and Dolff [46] for \((a, a, n)\)-tournaments is the special case \( a = b \) of Theorem 1. Theorem 3.1.4 of [22] is the special case \( a = b = 2 \). The theorem of Landau [55] is the special case \( a = b = 1 \) of Theorem 1.

**4.1 Definition of a testing algorithm**

The following algorithm INTERVAL-TEST decides whether a given \( D \) is a score sequence of an \((a, b, n)\)-tournament or not. This algorithm is based on Theorem 1 and returns \( W = \text{TRUE} \) if \( D \) is a score sequence, and returns \( W = \text{FALSE} \) otherwise.

<table>
<thead>
<tr>
<th>Player/Player</th>
<th>P_1</th>
<th>P_2</th>
<th>P_3</th>
<th>P_4</th>
<th>P_5</th>
<th>P_6</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>P_2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>P_3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>P_4</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>P_5</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>P_6</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Point matrix of a \((0,10,6)\)-tournament with \( f = 10 \) for \( D = (0,0,0,40,40,40) \).
\textit{Input.} \(a\): minimal number of points divided after each match; 
\(b\): maximal number of points divided after each match.

\textit{Output.} \(W\): logical variable (\(W = \text{true}\) shows that \(D\) is an \((a, b, n)\)-tournament.

\textit{Local working variables.} \(i\): cycle variable; 
\(L = (L_0, L_1, \ldots, L_n)\): the sequence of the values of the loss function.

\textit{Global working variables.} \(n\): the number of players \((n \geq 2)\); 
\(D = (d_1, d_2, \ldots, d_n)\): a nondecreasing sequence of nonnegative integers; 
\(B = (B_0, B_1, \ldots, B_n)\): the sequence of the binomial coefficients; 
\(S = (S_0, S_1, \ldots, S_n)\): the sequence of the sums of the \(i\) smallest scores.

\texttt{Interval-Test}(a, b)
\begin{verbatim}
01 for i ← 1 to n 
02 do \(L_i \leftarrow \max(L_{i-1}, bB_n - S_i - (n - i)d_i)\) 
03 if \(S_i < aB_i\) 
04 then \(W \leftarrow \text{false}\) 
05 return \(W\) 
06 if \(S_i > bB_n - L_i - (n - i)d_i\) 
07 then \(W \leftarrow \text{false}\) 
08 return \(W\) 
09 return \(W\)
\end{verbatim}

In worst case \texttt{Interval-Test} runs in \(\Theta(n)\) time even in the general case \(0 < a < b\) (\(n\) the best case the running time of \texttt{Interval-Test} is \(\Theta(n)\)). It is worth to mention, that the often referenced Havel–Hakimi algorithm \cite{32, 36} even in the special case \(a = b = 1\) decides in \(\Theta(n^2)\) time whether a sequence \(D\) is digraphical or not.

\subsection{Definition of an algorithm computing \(f\) and \(g\)}

The following algorithm is based on the bounds of \(f\) and \(g\) given by Lemma 3 and the logarithmic search algorithm described by D. E. Knuth \cite[page 410]{53}.

\textit{Input.} No special input (global working variables serve as input).

\textit{Output.} \(b\): \(f\) (the minimal \(F\)); 
\(a\): \(g\) (the maximal \(G\)).

\textit{Local working variables.} \(i\): cycle variable; 
\(l\): lower bound of the interval of the possible values of \(F\); 
\(u\): upper bound of the interval of the possible values of \(F\).
Global working variables. \( n \): the number of players (\( n \geq 2 \));
\( D = (d_1, d_2, \ldots, d_n) \): a nondecreasing sequence of nonnegative integers;
\( B = (B_0, B_1, \ldots, B_n) \): the sequence of the binomial coefficients;
\( S = (S_0, S_1, \ldots, S_n) \): the sequence of the sums of the \( i \) smallest scores;
\( W \): logical variable (its value is \( \text{TRUE} \), when the investigated \( D \) is a score sequence).

\( \text{MinF-MaxG} \)

01 \( B_0 \leftarrow S_0 \leftarrow L_0 \leftarrow 0 \) \hspace{1cm} \triangleright \text{Initialisation}
02 \text{for } i \leftarrow 1 \text{ to } n \\
03 \quad \text{do } B_i \leftarrow B_{i-1} + i - 1 \\
04 \quad S_i \leftarrow S_{i-1} + d_i \\
05 \quad l \leftarrow \max(\lceil S_n/B_n \rceil, \lceil d_n/(n-1) \rceil) \\
06 \quad u \leftarrow 2 \lceil d_n/(n-1) \rceil \\
07 \quad W \leftarrow \text{TRUE} \hspace{1cm} \triangleright \text{Computation of } f \\
08 \text{INTERVAL-Test}(0, l) \\
09 \quad \text{if } W = \text{TRUE} \\
10 \quad \text{then } b \leftarrow 1 \\
11 \quad \quad \quad \text{go to 21} \\
12 \quad b \leftarrow \lceil (l + u)/2 \rceil \\
13 \quad \text{INTERVAL-Test}(0, f) \\
14 \quad \text{if } W = \text{TRUE} \\
15 \quad \text{then go to 17} \\
16 \quad l \leftarrow b \\
17 \quad \text{if } u = l + 1 \\
18 \quad \text{then } b \leftarrow u \\
19 \quad \quad \quad \text{go to 21} \\
20 \quad \text{go to 14} \\
21 \quad l \leftarrow 0 \hspace{1cm} \triangleright \text{Computation of } g \\
22 \quad u \leftarrow f \\
23 \quad \text{INTERVAL-Test}(b, b) \\
24 \quad \text{if } W = \text{TRUE} \\
25 \quad \text{then } a \leftarrow f \\
26 \quad \quad \quad \text{go to 37} \\
27 \quad a \leftarrow \lceil (l + u)/2 \rceil \\
28 \quad \text{INTERVAL-Test}(0, a) \\
29 \quad \text{if } W = \text{TRUE} \\
30 \quad \text{then } l \leftarrow a \\
31 \quad \quad \quad \text{go to 33}
\begin{verbatim}
32 u ← a
33 if u = l + 1
34    then a ← l
35    go to 37
36 go to 27
37 return a, b

MinF-MaxG determines f and g.

Lemma 4 Algorithm MinG-MaxG computes the values f and g for arbitrary sequence D = (d_1, d_2, \ldots, d_n) in O(n \log(d_n/(n-1)/n)) time.

Proof. According to Lemma 3 F is an element of the interval \([d_n/(n-1)], [2d_n/(n-1)]\) and g is an element of the interval \([0, f]\). Using Theorem B of [53, page 412] we get that O(log(d_n/n)) calls of INTERVAL-TEST is sufficient, so the O(n) run time of INTERVAL-TEST implies the required running time of MinF-MaxG. ■

4.3 Computing of f and g in linear time

Analysing Theorem 1 and the work of algorithm MinF-MaxG one can observe that the maximal value of G and the minimal value of F can be computed independently by Linear-MinF-MaxG.

Input. No special input (global working variables serve as input).
Output. b: f (the minimal F).
a: g (the maximal G).
Local working variables. i: cycle variable.
Global working variables. n: the number of players (n \geq 2);
D = (d_1, d_2, \ldots, d_n): a nondecreasing sequence of nonnegative integers;
B = (B_0, B_1, \ldots, B_n): the sequence of the binomial coefficients;
S = (S_0, S_1, \ldots, S_n): the sequence of the sums of the i smallest scores.

Linear-MinF-MaxG
01 B_0 ← S_0 ← L_0 ← 0 ▷ Initialisation
02 for i ← 1 to n
03    do B_i ← B_{i-1} + i - 1
04    S_i ← S_{i-1} + d_i
05    a ← 0
06    b ← \min 2 \lfloor d_n/(n-1) \rfloor
07 for i ← 1 to n ▷ Computation of g
\end{verbatim}
Reconstruction of complete interval tournaments. II.

\[
\begin{align*}
08 & \quad \textbf{do } a_i \leftarrow \lceil 2S_i/(n^2 - n) \rceil \\
09 & \quad \textbf{if } a_i > a \quad \textbf{then } a \leftarrow a_i \\
10 & \quad \textbf{for } i \leftarrow 1 \text{ to } n \\
11 & \quad \textbf{do } L_i \leftarrow \max(L_{i-1}, bB_n - S_i - (n - i)d_i) \\
12 & \quad b_i \leftarrow (S_i + (n - i)d_i + L_i)/B_i \\
13 & \quad \textbf{if } b_i < b \quad \textbf{then } b \leftarrow b_i \\
14 & \textbf{return } a, b
\end{align*}
\]

Lemma 5. Algorithm \textsc{Linear-MinG-MaxG} computes the values \( f \) and \( g \) for arbitrary sequence \( D = (d_1, d_2, \ldots, d_n) \) in \( \Theta(n) \) time.

Proof. Lines 01–03, 07, and 18 require only constant time, lines 04–06, 09–12, and 13–17 require \( \Theta(n) \) time, so the total running time is \( \Theta(n) \).

5 Tournament with \( f \) and \( g \)

The following reconstruction algorithm \textsc{Score-Slicing2} is based on balancing between additional points (they are similar to ,,excess”, introduced by Brauer et al. [10]) and missing points introduced in [40]. The greediness of the algorithm Havel–Hakimi [32, 36] also characterises this algorithm.

This algorithm is an extended version of the algorithm \textsc{Score-Slicing} proposed in [40].

5.1 Definition of the minimax reconstruction algorithm

The work of the slicing program is managed by the following program \textsc{Mini-Max}.

Input. No special input (global working variables serve as input).

Output. \( M = [1 \ldots n, 1 \ldots n] \): the point matrix of the reconstructed tournament.

Local working variables. \( i, j \): cycle variables.

Global working variables. \( n \): the number of players \( (n \geq 2) \);
\( D = (d_1, d_2, \ldots, d_n) \): a nondecreasing sequence of nonnegative integers;
\( p = (p_0, p_1, \ldots, p_n) \): provisional score sequence;
\( P = (P_0, P_1, \ldots, P_n) \): the partial sums of the provisional scores;
\( M[1 \ldots n, 1 \ldots n] \): matrix of the provisional points.

\textsc{Mini-Max}
5.2 Definition of the score slicing algorithm

The key part of the reconstruction is the following algorithm SCORE-SLICING2 [40].

During the reconstruction process we have to take into account the following bounds:

\[
\begin{align*}
    a &\leq m_{i,j} + m_{j,i} \leq b \quad (1 \leq i < j \leq n); \\
    \text{modified scores have to satisfy (7);} \\
    m_{i,j} &\leq p_i \quad (1 \leq i, j \leq n, i \neq j); \\
    \text{the monotonicity } p_1 \leq p_2 \leq \ldots \leq p_k \text{ has to be saved} \quad (1 \leq k \leq n) \\
    m_{ii} &= 0 \quad (1 \leq i \leq n).
\end{align*}
\]

**Input.** \( k \): the number of the actually investigated players \( k > 2 \);
\( p_k = (p_0, p_1, p_2, \ldots, p_k) \) \( (k = 3, 4, \ldots, n) \): prefix of the provisional score sequence \( p \);
\( M[1 \ldots n, 1 \ldots n] \): matrix of provisional points.

**Output.** \( M[1 \ldots n, 1 \ldots n] \): matrix of provisional points;
\( p_k = (p_0, p_1, p_2, \ldots, p_k) \) \( (k = 2, 3, 4, \ldots, n-1) \): prefix of the provisional score sequence \( p \).

**Local working variables.** \( A = (A_1, A_2, \ldots, A_n) \): the number of the additional points;
Reconstruction of complete interval tournaments. II.

$M$: missing points (the difference of the number of actual points and the number of maximal possible points of $P_k$);
$d$: difference of the maximal decreasable score and the following largest score;
$y$: minimal number of sliced points per player;
$f$: frequency of the number of maximal values among the scores $p_1, p_2, \ldots, p_{k-1}$;
i, $j$: cycle variables;
m: maximal amount of sliceable points;
$P = (P_0, P_1, \ldots, P_n)$: the sums of the provisional scores;
x: the maximal index $i$ with $i < k$ and $m_{i,k} < b$.

Global working variables.
$n$: the number of players ($n \geq 2$);
$B = (B_0, B_1, B_2, \ldots, B_n)$: the sequence of the binomial coefficients;
a: minimal number of points divided after each match;
b: maximal number of points divided after each match.

Score-Slicing2($k, p_k, M$)

01 $P_0 \leftarrow 0$ ▷ Initialisation
02 for $i \leftarrow 1$ to $k - 1$
03 do $P_i \leftarrow P_{i-1} + p_i$
04 $A_i \leftarrow P_i - aB_i$
05 $M \leftarrow (k - 1)b - p_k$
06 while $M > 0$ and $A_{k-1} > 0$ ▷ There are missing and additional points
07 do $x \leftarrow k - 1$
08 while $r_{x,k} = b$
09 do $x \leftarrow x - 1$
10 $f \leftarrow 1$
11 while $p_{x-f+1} = p_{x-f}$
12 do $f \leftarrow f + 1$
13 $d \leftarrow p_{x-f+1} - p_{x-f}$
14 $m \leftarrow \min(b, d, \lceil A_x/f \rceil, \lceil M/f \rceil)$
15 for $i \leftarrow f$ downto 1
16 do $y \leftarrow \min(b - m_{x+1-i,k}, m, M, A_{x+1-i}, p_{x+1-i})$
17 $m_{x+1-i,k} \leftarrow m_{x+1-i,k} + y$
18 $p_{x+1-i} \leftarrow p_{x+1-i} - y$
19 $m_{k,x+1-i} \leftarrow m_{k,x+1-i} - m_{x+1-i,k}$
20 $M \leftarrow M - y$
21 for $j \leftarrow i$ downto 1
22 $A_{x+1-i} \leftarrow A_{x+1-i} - y$
23 while $M > 0$ and $A_{k-1} = 0$ ▷ No additional points
do for $i \leftarrow k - 1$ downto 1
  
  \begin{align*}
  y & \leftarrow \min(m_{k,i}, M, m_{k,i+m_{i,k}-a}) \\
  m_{kl} & \leftarrow m_{k,l} - y \\
  M & \leftarrow M - y
  \end{align*}

return $p_k, M$

Let's consider an example. Figure 2 shows the point table of a $(2, 10, 6)$-tournament $T$.

<table>
<thead>
<tr>
<th>Player/Player</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
<th>$P_5$</th>
<th>$P_6$</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>—</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$P_2$</td>
<td>1</td>
<td>—</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>$P_3$</td>
<td>3</td>
<td>3</td>
<td>—</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>19</td>
</tr>
<tr>
<td>$P_4$</td>
<td>8</td>
<td>2</td>
<td>5</td>
<td>—</td>
<td>2</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>$P_5$</td>
<td>9</td>
<td>9</td>
<td>5</td>
<td>7</td>
<td>—</td>
<td>2</td>
<td>32</td>
</tr>
<tr>
<td>$P_6$</td>
<td>8</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>—</td>
<td>34</td>
</tr>
</tbody>
</table>

Figure 2: The point table of a $(2, 10, 6)$-tournament $T$.

The score sequence of $T$ is $D = (9, 9, 19, 20, 32, 34)$. In [40] the algorithm SCORE-SLICING2 resulted the point table represented in Figure 3.

<table>
<thead>
<tr>
<th>Player/Player</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
<th>$P_5$</th>
<th>$P_6$</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>—</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>$P_2$</td>
<td>1</td>
<td>—</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>$P_3$</td>
<td>1</td>
<td>1</td>
<td>—</td>
<td>8</td>
<td>3</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>$P_4$</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>—</td>
<td>8</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>$P_5$</td>
<td>9</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>—</td>
<td>10</td>
<td>32</td>
</tr>
<tr>
<td>$P_6$</td>
<td>10</td>
<td>10</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>—</td>
<td>34</td>
</tr>
</tbody>
</table>

Figure 3: The point table of $T$ reconstructed by SCORE-SLICING2.

The algorithm MINI-MAX starts with the computation of $f$. MINF-MAXG called in line 01 begins with initialisation, including provisional setting of the elements of $\mathcal{M}$ so, that $m_{ij} = b$, if $i > j$, and $m_{ij} = 0$ otherwise. Then MINF-MAXG sets the lower bound $l = \max(9, 7) = 9$ of $f$ in line 05 and tests it in line 08 by INTERVAL-TEST. The test shows that $l = 9$ is large enough so MINI-MAX sets $b = 9$ in line 12 and jumps to line 21 and begins to compute $g$. INTERVAL-TEST called in line 23 shows that $a = 9$ is too large, therefore
MINF-MAXG continues with the test of $a = 5$ in line 27. The result is positive, therefore comes the test of $a = 7$, then the test of $a = 8$. Now $u = 1 + 1$ in line 33, so $a = 8$ is fixed, and the control returns to line 02 of MINI-MAX.

Lines 02–08 contain initialisation, and MINI-MAX begins the reconstruction of a $(8,9,6)$-tournament in line 9. The basic idea is that MINI-MAX successively determines the won and lost points of $P_6$, $P_5$, $P_4$ and $P_3$ by repeated calls of SCORE-SLICING2 in line 11, and finally it computes directly the result of the match between $P_2$ and $P_1$ in lines 12–14.

At first MINI-MAX computes the results of $P_6$ calling SCORE-SLICING2 with parameter $k = 6$. The number of additional points of the first five players is $A_5 = 89 - 8 \cdot 10 = 9$ according to line 04, the number of missing points of $P_6$ is $M = 5 \cdot 9 - 34 = 11$ according to line 05. Then SCORE-SLICING2 determines the number of maximal numbers among the provisional scores $p_1, p_2, \ldots, p_5$ ($f = 1$ according to lines 10–12) and computes the difference between $p_5$ and $p_4$ ($d = 12$ according to line 13). In line 14 we get, that $m = 9$ points are sliceable, and $P_5$ gets these points in the match with $P_6$ in line 17, so the number of missing points of $P_6$ decreases to $M = 11 - 9 = 2$ (line 20) and the number of additional point decreases to $A_5 = 9 - 9 = 0$. Therefore the computation continues in lines 23–28 and $m_{64}$ and $m_{63}$ will be decreased by 1 resulting $m_{64} = 8$ and $m_{63} = 8$ as the seventh line and seventh column of Figure 4 show. The returned score sequence is $p_5 = (9, 9, 19, 20, 23)$.

<table>
<thead>
<tr>
<th>Player/Player</th>
<th>P₁</th>
<th>P₂</th>
<th>P₃</th>
<th>P₄</th>
<th>P₅</th>
<th>P₆</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td></td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>P₂</td>
<td>4</td>
<td></td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>P₃</td>
<td>4</td>
<td>4</td>
<td></td>
<td>7</td>
<td>4</td>
<td>0</td>
<td>19</td>
</tr>
<tr>
<td>P₄</td>
<td>7</td>
<td>7</td>
<td>1</td>
<td></td>
<td>5</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>P₅</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td></td>
<td>9</td>
<td>32</td>
</tr>
<tr>
<td>P₆</td>
<td>9</td>
<td>9</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td></td>
<td>34</td>
</tr>
</tbody>
</table>

Figure 4: The point table of $T$ reconstructed by MINI-MAX.

Second time MINI-MAX calls SCORE-SLICING2 with parameter $k = 5$, and get $A_4 = 9$ and $M = 13$. At first $P_4$ gets 1 point, then $P_3$ and $P_4$ get both 4 points, reducing $M$ to 4 and $A_4$ to 0. The computation continues in line 23 and results the further decrease of $m_{54}$, $m_{53}$, $m_{52}$, and $m_{51}$ by 1, resulting $m_{54} = 3$, $m_{53} = 4$, $m_{52} = 8$, and $m_{51} = 8$ as the sixth row of Figure 4 shows. The returned score sequence is $p_4 = (9, 9, 15, 15)$.
Third time Mini-Max calls Score-Slicing2 with parameter $k = 4$, and get $A_3 = 11$ and $M = 11$. At first $P_3$ gets 6 points, then $P_3$ further 1 point, and $P_2$ and $P_1$ also both get 1 point, resulting $m_{34} = 7, m_{43} = 2, m_{42} = 8, m_{24} = 1, m_{14} = 1$ and $m_{14} = 8$, further $A_3 = 0$ and $M = 2$. The computation continues in lines 23–28 and results a decrease of $m_{43}$ by 1 point resulting $m_{43} = 1, m_{42} = 7$, and $m_{41} = 7$, as the fifth row and fifth column of Figure 4 show. The returned score sequence is $p_3 = (8, 8, 8)$.

Fourth time Mini-Max calls Score-Slicing2 with parameter $k = 3$, and gets $A_2 = 8$ and $M = 10$. At first $P_1$ and $P_2$ get 4 points, resulting $m_{13} = 4$, and $m_{23} = 4$, and $M = 2$, and $A_2 = 0$. Then Mini-Max sets in lines 23–26 $m_{31} = 4$ and $m_{32} = 4$. The returned score sequence is $p_2 = (4, 4)$.

Finally Mini-Max sets $m_{12} = 4$ and $m_{21} = 4$ in lines 14–15 and returns the point matrix represented in Figure 4.

The comparison of Figures 3 and 4 shows a large difference between the simple reconstruction of Score-Slicing2 and the minimax reconstruction of Mini-Max: while in the first case the maximal value of $m_{ij} + m_{ji}$ is 10 and the minimal value is 2, in the second case the maximum equals to 9 and the minimum equals to 8, that is the result is more balanced (the given $D$ does not allow to build a perfectly balanced $(k, k, n)$-tournament).

5.3 Analysis of the minimax reconstruction algorithm

The main result of this paper is the following assertion.

**Theorem 2** If $n \geq 2$ is a positive integer and $D = (d_1, d_2, \ldots, d_n)$ is a non-decreasing sequence of nonnegative integers, then there exist positive integers $f$ and $g$, and a $(g, f, n)$-tournament $T$ with point matrix $M$ such, that

$$f = \min(m_{ij} + m_{ji}) \leq b, \quad (14)$$

$$g = \max m_{ij} + m_{ji} \geq a \quad (15)$$

for any $(a, b, n)$-tournament, and algorithm Linear-MinF-MaxG computes $f$ and $g$ in $\Theta(n)$ time, and algorithm Mini-Max generates a suitable $T$ in $O(d_n n^2)$ time.

**Proof.** The correctness of the algorithms Score-Slicing2, MinF-MaxG implies the correctness of Mini-Max.

Lines 1–46 of Mini-Max require $O(\log(d_n / n))$ uses of MING-MaxF, and one search needs $O(n)$ steps for the testing, so the computation of $f$ and $g$ can be executed in $O(n \log(d_n / n))$ times.
The reconstruction part (lines 47–55) uses algorithm SCORE-SLICING2, which runs in $O(bn^3)$ time [40]. MINI-MAX calls SCORE-SLICING2 $n - 2$ times with $f \leq 2[d_n/n]$, so $n^3d_n/n = d_n n^2$ finishes the proof.

The property of the tournament reconstruction problem that the extremal values of $f$ and $g$ can be determined independently and so there exists a tournament $T$ having both extremal features is called linking property. This concept was introduced by Ford and Fulkerson in 1962 [17] and later extended by A. Frank in [22].

6 Summary

A nondecreasing sequence of nonnegative integers $D = (d_1, d_2, \ldots, d_n)$ is a score sequence of a $(1, 1, 1)$-tournament, iff the sum of the elements of $D$ equals to $B_n$ and the sum of the first $i$ ($i = 1, 2, \ldots, n-1$) elements of $D$ is at least $B_i$ [55].

$D$ is a score sequence of a $(k, k, n)$-tournament, iff the sum of the elements of $D$ equals to $kB_n$, and the sum of the first $i$ elements of $D$ is at least $kB_i$ [46, 60].

$D$ is a score sequence of an $(a, b, n)$-tournament, iff (7) holds [40].

In all 3 cases the decision whether $D$ is digraphical requires only linear time.

In this paper the results of [40] are extended proving that for any $D$ there exists an optimal minimax realization $T$, that is a tournament having $D$ as its out-degree sequence, and maximal $G$, and minimal $F$ in the set of all realizations of $D$.

In a continuation [41] of this paper we construct balanced as possible tournaments in a similar way if not only the out-degree sequence but the in-degree sequence is also given.

Acknowledgement

The author thanks András Frank (Eötvös Loránd University) for valuable advises concerning the application of flow theory, Péter L. Erdős (Alfréd Rényi Institute of Mathematics of HAS) for the consultation and the unknown referee proposing corrections of the manuscript.

The research was supported by the project TÁMOP-4.2.1.B-09/1/KMR–2010-003 of Eötvös Loránd University.
References


Received: March 5, 2010; Revised: April 18, 2010