

# Algorithms for Finding Link-Irregular Tournaments

Alexander Bastien

## Introduction

This document provides detailed algorithmic descriptions for the computational search methods used to find link-irregular tournaments, as referenced in our paper on the existence of link-irregular tournaments. A tournament  $T$  is *link-irregular* if no two distinct vertices have isomorphic link graphs, where the link graph  $L(v)$  of a vertex  $v$  is the induced subgraph on  $N^-(v) \cup N^+(v)$  (the union of in-neighbors and out-neighbors of  $v$ ).

Our approach employs a multi-strategy search combining random generation, hill-climbing optimization, and seeded extension techniques. The algorithms use signature-based filtering to efficiently detect link graph isomorphisms, followed by complete isomorphism verification using the VF2 algorithm. These methods successfully found link-irregular tournaments for all tested values  $n \in \{6, 7, \dots, 100\}$ , providing strong computational evidence for our conjecture on the existence of link-irregular tournaments.

The complete Python implementation of these algorithms is available at: <https://github.com/yourusername/link-irregular-tournaments>

## Algorithm Descriptions

The following algorithms implement our search strategy:

- **Algorithm 1:** Random search baseline
- **Algorithm 2:** Hill-climbing optimization
- **Algorithm 3:** Seeded extension
- **Algorithm 4:** Collision detection
- **Algorithm 5:** Signature computation
- **Algorithm 6:** Main multi-strategy search

# 1 Algorithms

---

**Algorithm 1** Random Search for Link-Irregular Tournaments

---

**Require:** Number of vertices  $n$ , maximum attempts  $k$

**Ensure:** Link-irregular tournament  $T$  or NULL

```
1: for  $t = 1$  to  $k$  do  
2:    $T \leftarrow \text{RANDOMTOURNAMENT}(n)$   
3:    $C \leftarrow \text{COMPUTECOLLISIONS}(T)$   
4:   if  $C = \emptyset$  then  
5:     return  $T$   
6:   end if  
7: end for  
8: return NULL
```

---

---

**Algorithm 2** Hill-Climbing Search for Link-Irregular Tournaments

---

**Require:** Number of vertices  $n$ , steps  $s$ , restarts  $r$

**Ensure:** Link-irregular tournament  $T$  or best found tournament

```
1:  $T_{\text{best}} \leftarrow \text{NULL}$ 
2:  $|C_{\text{best}}| \leftarrow \infty$ 
3: for  $i = 1$  to  $r$  do
4:    $T \leftarrow \text{RANDOMTOURNAMENT}(n)$ 
5:    $C \leftarrow \text{COMPUTECOLLISIONS}(T)$ 
6:   if  $C = \emptyset$  then
7:     return  $T$ 
8:   end if
9:   for  $j = 1$  to  $s$  do
10:    if  $C = \emptyset$  then
11:      return  $T$ 
12:    end if
13:     $(u, v) \leftarrow$  random element from  $C$ 
14:     $E_{\text{cand}} \leftarrow \{e \in E(T) : e \text{ incident to } u \text{ or } v\}$ 
15:    Sample up to 20 edges from  $E_{\text{cand}}$ 
16:     $e^* \leftarrow \text{NULL}, |C^*| \leftarrow |C|$ 
17:    for each sampled edge  $e = \{a, b\}$  do
18:      Flip orientation of  $e$  in  $T$ 
19:       $C' \leftarrow \text{COMPUTECOLLISIONS}(T)$ 
20:      if  $|C'| = 0$  then
21:        return  $T$ 
22:      end if
23:      if  $|C'| < |C^*|$  then
24:         $e^* \leftarrow e, |C^*| \leftarrow |C'|$ 
25:      end if
26:      Revert flip of  $e$ 
27:    end for
28:    if  $e^* \neq \text{NULL}$  then
29:      Flip orientation of  $e^*$  in  $T$ 
30:       $C \leftarrow \text{COMPUTECOLLISIONS}(T)$ 
31:    end if
32:    if  $|C| < |C_{\text{best}}|$  then
33:       $T_{\text{best}} \leftarrow T, C_{\text{best}} \leftarrow C$ 
34:    end if
35:  end for
36: end for
37: return  $T_{\text{best}}$ 
```

---

---

**Algorithm 3** Seeded Extension Search

---

**Require:** Target size  $n$ , seed tournament  $T_0$  on  $k < n$  vertices, attempts  $m$

**Ensure:** Link-irregular tournament  $T$  or NULL

```
1: for  $a = 1$  to  $m$  do
2:    $T \leftarrow T_0$ 
3:   for  $v = k$  to  $n - 1$  do
4:     Add vertex  $v$  to  $T$ 
5:     for each  $u < v$  do
6:       Add edge  $(u, v)$  or  $(v, u)$  to  $T$  with equal probability
7:     end for
8:   end for
9:    $C \leftarrow \text{COMPUTECOLLISIONS}(T)$ 
10:  if  $C = \emptyset$  then
11:    return  $T$ 
12:  end if
13:   $T' \leftarrow \text{HILLCLIMB}(T, 1500, 1)$  ▷ Local refinement
14:   $C' \leftarrow \text{COMPUTECOLLISIONS}(T')$ 
15:  if  $C' = \emptyset$  then
16:    return  $T'$ 
17:  end if
18: end for
19: return NULL
```

---

---

**Algorithm 4** Compute Link Collisions

---

**Require:** Tournament  $T = (V, E)$

**Ensure:** Set of collision pairs  $C$

```
1:  $C \leftarrow \emptyset$ 
2:  $\mathcal{S} \leftarrow$  empty map from signatures to vertex lists
3: for each  $v \in V$  do
4:    $L_v \leftarrow$  induced subgraph on  $N^-(v) \cup N^+(v)$ 
5:    $\sigma_v \leftarrow \text{SIGNATURE}(L_v)$  ▷  $(|V|, |E|, \text{deg-seq}, \tau_3)$ 
6:   Append  $v$  to  $\mathcal{S}[\sigma_v]$ 
7: end for
8: for each signature  $\sigma$  with  $|\mathcal{S}[\sigma]| \geq 2$  do
9:   for each pair  $(u, v)$  with  $u, v \in \mathcal{S}[\sigma]$ ,  $u < v$  do
10:    if  $\text{IsISOMORPHIC}(L_u, L_v)$  then
11:       $C \leftarrow C \cup \{(u, v)\}$ 
12:    end if
13:   end for
14: end for
15: return  $C$ 
```

---

---

**Algorithm 5** Link Graph Signature

---

**Require:** Directed graph  $L = (V, E)$ **Ensure:** Invariant signature  $\sigma$ 

```
1:  $n \leftarrow |V|$ ,  $m \leftarrow |E|$ 
2:  $D \leftarrow$  sorted list of  $(\text{in-deg}(v), \text{out-deg}(v))$  for all  $v \in V$ 
3:  $\tau_3 \leftarrow 0$ 
4: for each triple  $(a, b, c)$  of distinct vertices do
5:   if  $(a, b), (b, c), (c, a) \in E$  or  $(a, c), (c, b), (b, a) \in E$  then
6:      $\tau_3 \leftarrow \tau_3 + 1$ 
7:   end if
8: end for
9: return  $(n, m, D, \tau_3)$ 
```

---

---

**Algorithm 6** Multi-Strategy Search (Main Algorithm)

---

**Require:** Number of vertices  $n$ **Ensure:** Link-irregular tournament  $T$  or NULL

```
1:  $T \leftarrow \text{RANDOMSEARCH}(n, 300)$ 
2: if  $T \neq \text{NULL}$  then
3:   return  $T$ 
4: end if
5:  $T \leftarrow \text{HILLCLIMB}(n, 6000, 5)$ 
6: if  $T \neq \text{NULL}$  then
7:   return  $T$ 
8: end if
9:  $T_0 \leftarrow$  known link-irregular tournament on 6 vertices
10:  $T \leftarrow \text{SEEDEDEXTENSION}(n, T_0, 50)$ 
11: return  $T$ 
```

---