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 $n \in \{6, 7, \ldots, 100\}$, 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/omidkhormali/Link-irregular-Digraphs/tree/main`

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

for $t = 1$ to k **do**
  $T \leftarrow \textsc{RandomTournament}(n)$
  $C \leftarrow \textsc{ComputeCollisions}(T)$
  if $C = \emptyset$ **then**
    return T
  end if
end for
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

$T_{\text{best}} \leftarrow \textsc{null}$
$|C_{\text{best}}| \leftarrow \infty$
for $i = 1$ to r **do**
    $T \leftarrow \textsc{RandomTournament}(n)$
    $C \leftarrow \textsc{ComputeCollisions}(T)$
    if $C = \emptyset$ **then**
        return T
    end if
    for $j = 1$ to s **do**
        if $C = \emptyset$ **then**
            return T
        end if
        $(u, v) \leftarrow$ random element from C
        $E_{\text{cand}} \leftarrow \{e \in E(T) : e$ incident to u or $v\}$
        Sample up to 20 edges from E_{cand}
        $e^* \leftarrow \textsc{null}$, $|C^*| \leftarrow |C|$
        for each sampled edge $e = \{a, b\}$ **do**
            Flip orientation of e in T
            $C' \leftarrow \textsc{ComputeCollisions}(T)$
            if $|C'| = 0$ **then**
                return T
            end if
            if $|C'| < |C^*|$ **then**
                $e^* \leftarrow e$, $|C^*| \leftarrow |C'|$
            end if
            Revert flip of e
        end for
        if $e^* \neq \textsc{null}$ **then**
            Flip orientation of e^* in T
            $C \leftarrow \textsc{ComputeCollisions}(T)$
        end if
        if $|C| < |C_{\text{best}}|$ **then**
            $T_{\text{best}} \leftarrow T$, $C_{\text{best}} \leftarrow C$
        end if
    end for
end for
return T_{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

for $a = 1$ to m **do**
  $T \leftarrow T_0$
  for $v = k$ to $n - 1$ **do**
    Add vertex v to T
    for each $u < v$ **do**
      Add edge (u, v) or (v, u) to T with equal probability
    end for
  end for
  $C \leftarrow$ COMPUTECOLLISIONS(T)
  if $C = \emptyset$ **then**
    return T
  end if
  $T' \leftarrow$ HILLCLIMB(T, 1500, 1) ▷ Local refinement
  $C' \leftarrow$ COMPUTECOLLISIONS(T')
  if $C' = \emptyset$ **then**
    return T'
  end if
end for
return NULL

Algorithm 4 Compute Link Collisions

Require: Tournament $T = (V, E)$
Ensure: Set of collision pairs C

$C \leftarrow \emptyset$
$\mathcal{S} \leftarrow$ empty map from signatures to vertex lists
for each $v \in V$ **do**
  $L_v \leftarrow$ induced subgraph on $N^-(v) \cup N^+(v)$
  $\sigma_v \leftarrow$ SIGNATURE(L_v) ▷ $(|V|, |E|, \text{deg-seq}, \tau_3)$
  Append v to $\mathcal{S}[\sigma_v]$
end for
for each signature σ with $|\mathcal{S}[\sigma]| \geq 2$ **do**
  for each pair (u, v) with $u, v \in \mathcal{S}[\sigma]$, $u < v$ **do**
    if ISISOMORPHIC(L_u, L_v) **then**
      $C \leftarrow C \cup \{(u, v)\}$
    end if
  end for
end for
return C

Algorithm 5 Link Graph Signature

Require: Directed graph $L = (V, E)$
Ensure: Invariant signature σ

$n \leftarrow |V|$, $m \leftarrow |E|$
$D \leftarrow$ sorted list of $(\text{in-deg}(v), \text{out-deg}(v))$ for all $v \in V$
$\tau_3 \leftarrow 0$
for each triple (a, b, c) of distinct vertices **do**
  if $(a, b), (b, c), (c, a) \in E$ **or** $(a, c), (c, b), (b, a) \in E$ **then**
    $\tau_3 \leftarrow \tau_3 + 1$
  end if
end for
return (n, m, D, τ_3)

Algorithm 6 Multi-Strategy Search (Main Algorithm)

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

$T \leftarrow$ RandomSearch(n, 300)
if $T \neq$ NULL **then**
  return T
end if
$T \leftarrow$ HillClimb(n, 6000, 5)
if $T \neq$ NULL **then**
  return T
end if
$T_0 \leftarrow$ known link-irregular tournament on 6 vertices
$T \leftarrow$ SeededExtension(n, T_0, 50)
return T