
André Nichterlein

Algorithmics and Computational Complexity, Faculty IV, TU Berlin

Shonan Meeting 144, March 5th

2-Club Problem

Input: An undirected graph $G = (V, E)$.

Task: Find the maximum size 2-club (diameter-two subgraph) in $G$.

- proposed as clique relaxation in social network analysis [Mokken; Quality and Quantity, 1979]
- NP-hard [Balasundaram, Butenko, Trukhanov; Journal of Combinatorial Optimization, 2005]
- NP-hard to approximate within a factor $|V|^{1/2-\epsilon}$ [Asahiro, Doi, Miyano, Samizo, Shimizu; Algorithmica, 2018]
2-Club is hard? Not in practice!

Existing implementation: [Hartung, Komusiewicz, N.; Journal of Graph Algorithms and Applications, 2015]

**Data set:** Graphs from **clustering** and **coauthor** category of the 10th DIMACS challenge

**Implementation:** Written in Java Machine: CPU 3.60 GHz (Xeon); 64 GB main memory

---

![Graph size vs. Time for n](image1)

![Graph size vs. Time for m](image2)
Analysis: 2-Club size

Almost optimal algorithm:
Return a maximum degree vertex with its neighbors

Komusiewicz, Nichterlein, Niedermeier, Picker
Well-Connected 2-Club

1.1 2-Club
2-Club extensions

**Definition ([Veremyev, Boginski; Eur J Oper Res, 2012])**

**t-robust 2-club** $G$:
Any pair of vertices is connected by $t$ internally vertex-disjoint paths of length at most two.

**Definition ([Pattillo, Youssef, Butenko; Eur J Oper Res, 2013])**

**t-hereditary 2-club** $G$:
$G - U$ is a 2-club for all $U \subset V(G)$ where $|U| \leq t$.
$\iff$ any pair of nonadjacent vertices has $t + 1$ common neighbors.

**Definition ([Pattillo, Youssef, Butenko; Eur J Oper Res, 2013])**

**t-connected 2-club** $G$:
$G$ is a 2-club and $t$-vertex-connected.

**Example:** A $K_{3,3}$ is a
- 1-robust 2-club
- 2-hereditary 2-club
- 3-connected 2-club
Our results

**Goal:**
transfer algorithmic work (theoretical & practical) from 2-Club to $t$-robust / $t$-hereditary / $t$-connected 2-Club

**Results:**
- “unifying view” on all three considered models
- FPT algorithms
- competitive implementation
Simple search tree

Example: Find largest 2-hereditary 2-club (deleting any 2 vertices yields a 2-club)

Observation: At most one red vertex in a solution.

Generic Search tree:
FindSolution($G$)
1. If $G$ is a solution then return $G$
2. $u, v \leftarrow$ two “incompatible” vertices
3. Return $\max\{\text{FindSolution}(G - v), \text{FindSolution}(G - u)\}$

\[ \leadsto \text{running time } O(2^\ell nm) \]

Note: no $2^{(1-\varepsilon)\ell n^{O(1)}}$ algorithm for any $\varepsilon > 0$, unless SETH fails

Compatibile vertices — unifying view

Definition

Two vertices $v$ and $w$ in a graph are called **compatible**

- for **$t$-robust 2-clubs** if they are adjacent and have at least $t - 1$ common neighbors, or if they have at least $t$ common neighbors,
- for **$t$-hereditary 2-clubs** if they are adjacent or if they have at least $t + 1$ common neighbors,
- for **$t$-connected 2-clubs** if they are at distance at most two and are connected by at least $t$ internally vertex-disjoint paths.
1. sol ← ∅
2. foreach $v \in V$ do
3. \[T \leftarrow \text{all vertices at distance } \geq 2 \text{ from } v\]
4. \[S \leftarrow \text{largest solution in } T \text{ that contains } v\]
5. if $S$ is larger than sol then sol ← $S$
6. delete $v$
7. return sol
Turing kernelization — practical effect

**Advantage:**
Turing kernelization allows to store data for each pair of vertices (e.g. number of common neighbors)
Data reduction & lower bounds

Reduction Rule
Remove vertices whose degree is too low.

Incompatibility graph:
Two vertices are adjacent in the the *incompatibility graph* iff they are not compatible.

Observation:
The size of a maximum independent set in the *incompatibility graph* is an upper bound on the solution size in the input graph.
⇒ upper bound worse than best previously found solution ⇒ discard current Turing kernel
Experiments I

Data set: Graphs from clustering and coauthor category of the 10th DIMACS challenge

Implementation: Written in Java

Machine: CPU 3.60 GHz (Xeon); 64 GB main memory

2-Club implementations:

- **HKN** (Java)

- **CHLS** (c++)
  [Chang, Hung, Lin, Su; Computing, 2013.]
Experiments II

Data set: Graphs from clustering and coauthor category of the 10th DIMACS challenge

Implementation: Written in Java

Machine: CPU 3.60 GHz (Xeon); 64 GB main memory

Our Implementation

Their Implementation

Implementations:

2-connected 2-Club

- **BB** - Branch & Bound (c++)
- **ILP** (c++ & Gurobi)

[Yezerska, Pajouh, Butenko; European Journal of Operational Research, 2017.]
Experiments III

Graph: coPapersCiteeseer

Running time\((t - 1)\)-hereditary \(t\)-robust
\(t\)-connected

Graph: coPapersCiteeseer

Experimental Results
Experiments IV

Graph: coAuthorsCiteseer

- $(t - 1)$-hereditary
- $t$-robust
- $t$-connected

Running time vs. $t$

2-club size vs. $t$
Summary & Outlook

Key results:
- Unifying approach for several 2-club variants.
- Efficient implementation (= data reduction + Turing kernelization + search tree).

Work in progress: $\gamma$-relative robust 2-club $S$:
$0 < \gamma \leq 1$: Any pair of vertices connected by at least $\gamma \cdot |S|$ paths of length at most two.

Example: $\gamma = 0.5$

Open Question: Is $t$-robust / $t$-hereditary / $t$-connected 2-club fixed parameter tractable with respect to the solution size?

Thank you!