# Not Clique-Bait: the Bron-Kerbosch Algorithm as a Solution to the Maximum Clique Problem

Sophie Quinn, Brendan Banfield, Aaron Banse, Kellen Knop | Advisor: Layla Oesper
Carleton College Department of Computer Science

Carleton

## Terminology

**Graph**: a set of nodes and the edges between them
**Clique**: a subset of nodes in a graph in which every node connects to every other node
**Maximum Clique**: the largest clique in a given graph
**Maximal Clique**: a clique that cannot be expanded by adding any other node

## Methodology

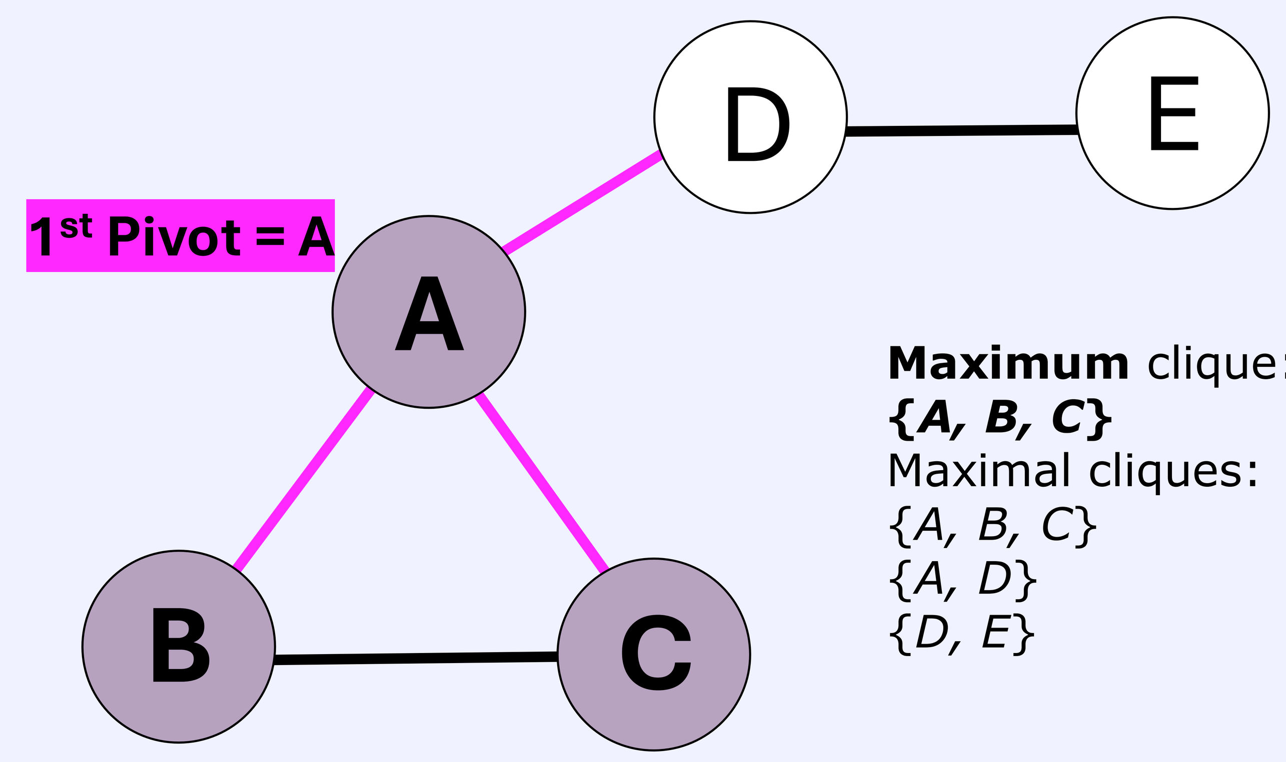
- Our group **implemented** 4 algorithms that solve the maximum clique problem
- I **focused** on the Bron-Kerbosch algorithm [1] [2]
  - **GOAL**: Explore the runtime of Bron-Kerbosch (pivot and non-pivot versions) on different graphs, accounting for # of nodes and edge density
- **Tested** on randomly-generated graphs

## The Algorithm

- **Recursive** and exact
- Built to find all **maximal** cliques
  - Modified to only return **maximum** clique
- 2 versions: non-pivot and pivot
  - Focused on pivot: more efficient optimization
- How it works:
  - *Tries different combinations of nodes to find all maximal cliques*
  - Maintains 3 sets of nodes: R (current combination), P (candidates to add to R), X (already tried in R)
  - **Our Goal**: Maximize # nodes in R

```
Function BronKerbosch(R, P, X):
    if P ∪ X == ∅ then
        Report R as clique;
        return
    end
    Choose pivot u ∈ (P ∪ X) with max(N(u) ∩ P) ;
    for v in P/N(u) do
        BronKerbosch(R ∪ v, P ∩ N(v), X ∩ N(v));
        P = P/v;
        X = X ∪ v;
    end
```

## An Example




**Maximum** clique:
***{A, B, C}***
Maximal cliques:
*{A, B, C}*
*{A, D}*
*{D, E}*

1. {}{A, B, C, D, E}{}
2. **pivot = A**
3. → {A}{B, C, D}{}
4. → pivot = B
5. → {A, B}{C}{}
6. → pivot = C
7. **{A, B, C}{}{}** return maximal
8. {A}{D}{B}
9. → {A, D}{}{}
10. → **{A, D}{}{}** return maximal
11. {}{B, C, D, E}{A}
12. → {E}{D}{}
13. → pivot = D
14. **{E, D}{}{}** return maximal

**Legend**
{…}{…}{…} = {R}{P}{X} = {current}{candidates}{tried}
→ = 1 iteration of the for-loop
→ = 1 recursive step

## Results

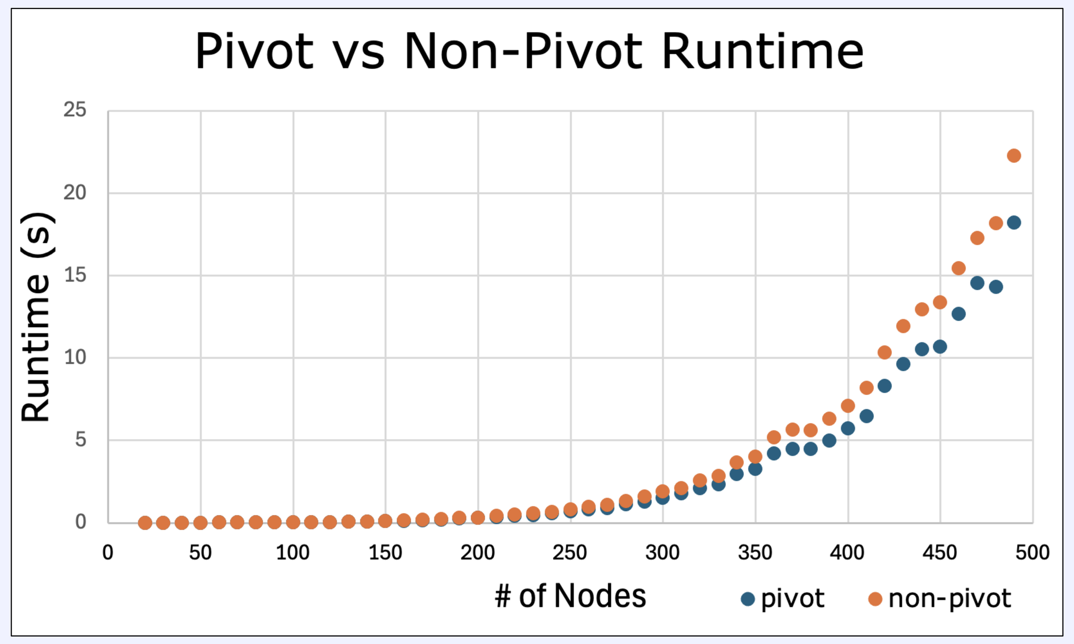



Figure 1. Median runtimes of pivot vs non-pivot solvers on 10 graphs each of sizes [20, 30, …, 490]. Average edge density 0.387 (ranging from ~0.3 to ~0.4).

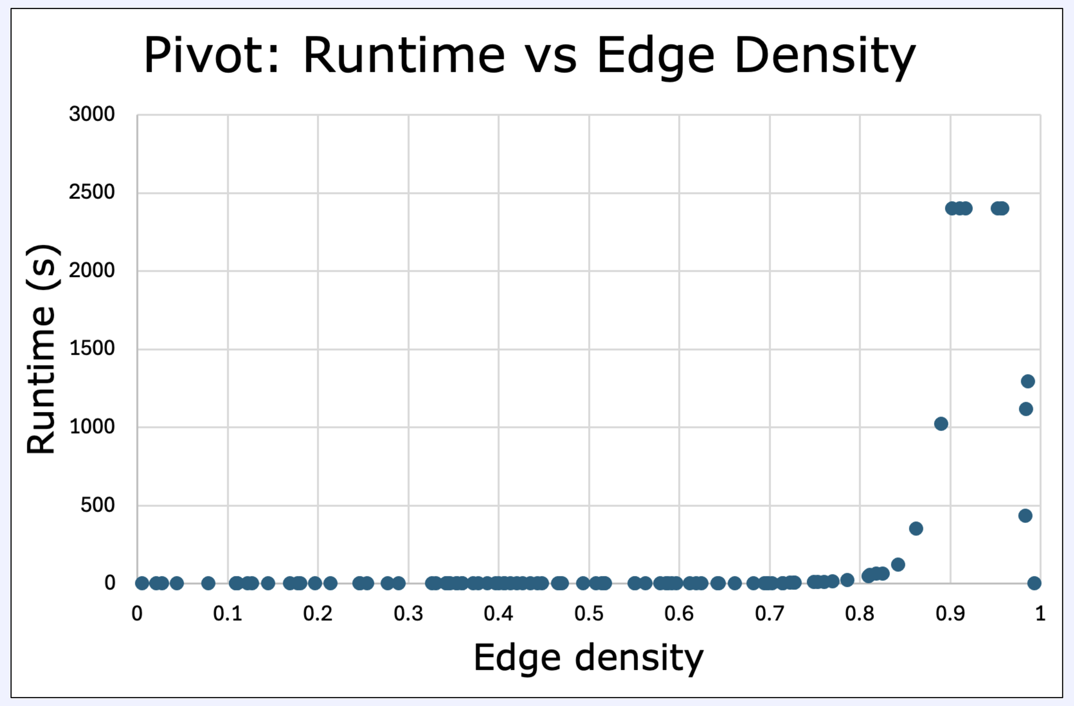



Figure 2. Runtime of pivot solver on 100 graphs of size 100 with increasing edge density ( $\frac{E}{N(N-1)}$ ). Time cutoff of 2400 seconds (40 min).

## Conclusions

- Is indeed exponential. Pivot is indeed faster than non-pivot
- Worst-case edge density is approximately range [0.9, 0.95]
- Future work:
  - Compare pivot and non-pivot versions at different edge densities
  - Compare more in-depth with branch-and-bound (our other exact algorithm) at varying edge densities

## References

[1] Bron, Coen, and Joep Kerbosch. "Algorithm 457: Finding All Cliques of an Undirected Graph." 1973
[2] Cazals, F., and C. Karande. "A Note on the Problem of Reporting Maximal Cliques." 2008.

## Acknowledgements

Thank you to Layla for her support and feedback throughout the term. Thank you to Mike Tie for technical support.