# COMS4995 Parallel Functional Programming Project Proposal
# Clique Problem with Bron–Kerbosch Algorithm

Xuezhen Wang, Songheng Yin

Uni: ww2604, sy3079

## 1 Overview

The project aims to implement the Bron-Kerbosch algorithm using Haskell to identify all maximal cliques within an arbitrary undirected graph.

The implementation will include both a single-threaded sequential version and a parallelized counterpart. Our objective is to analyze their performance to assess the impact and effectiveness of parallelism in this context.

## 2 Background

### 2.1 Definitions

In an undirected graph $G = (V, E)$. A **clique** $C$ is a subset of the vertices, $C \subseteq V$, such that every two distinct vertices are adjacent, that is, the induced subgraph by $C$ is a complete graph $K_{|C|}$.

A **maximal clique** is a clique that cannot be extended by including one more adjacent vertex, that is, a clique which does not exist exclusively within the vertex set of a larger clique.

A **maximum clique** of a graph is a clique such that there is no clique with more vertices. Moreover, the clique number $\omega(G)$ of a graph $G$ is the number of vertices in a maximum clique in $G$.

Note the difference between maximal clique and maximum clique: a maximum clique is always maximal, the converse is not always true.

### 2.2 Problem

The clique decision problem asking for if a clique of size $k$ exists in the given graph. It was one of Richard Karp's original 21 problems shown NP-complete in Cook [1971] and Karp [1972]

Our task, a variant of the clique decision problem, is to list **all** the **maximal** cliques given an undirected graph $G$.

It is easy to know the problem is impossible to be done in polynomial running time since it can derive the answer to the clique decision problem trivially.

## 3 Algorithm

The Bron–Kerbosch algorithm, designed by Bron and Kerbosch [1973], is an enumeration algorithm for finding all maximal cliques in an undirected graph.

**function** BronKerbosch($R, P, X$)
    **if** $P$ and $X$ are both empty **then**
        Report $R$ as a maximal clique ▷ A maximal clique is found
    **end if**
    **for** each vertex $v$ in $P$ **do**
        BronKerbosch($R \cup \{v\}, P \cap N(v), X \cap N(v)$) ▷ Explore extensions of $R$ including $v$
        $P \leftarrow P \setminus \{v\}$ ▷ Remove $v$ from potential clique extensions
        $X \leftarrow X \cup \{v\}$ ▷ Add $v$ to excluded set for this recursion level
    **end for**
**end function**

The Bron-Kerbosch algorithm uses three sets $R$, $P$, and $X$ to find maximal cliques in an undirected graph:

1. $R$ (Reported Clique): This set starts empty and grows as the algorithm progresses. It represents the current clique being constructed. When both $P$ and $X$ are empty, $R$ is a maximal clique and is reported as such.

2. $P$ (Potential Nodes): This set contains vertices that are connected to all vertices in $R$ and might be included in the clique. These are potential candidates to be added to $R$. The algorithm iteratively moves vertices from $P$ to $R$ to explore the expansion of the current clique.

3. $X$ (Excluded Nodes): This set also starts empty and contains vertices that have been considered for inclusion in $R$ and found not to lead to a maximal clique (in the current path of the search). It helps to avoid re-examining the same vertex within the same recursive call.

## 4 Objectives and Workflow

### 4.1 Experiment Preparation

We will use a script to generate a large dataset including both random data and corner cases as the input to our Haskell program. The expected solution is generated simultaneously and can be used later to verify the correctness of our algorithm implementation.

### 4.2 Experiment Design

We are going to run the naive sequential and parallel algorithms respectively and monitor their performances. To ensure the effect of parallelism with GHC, we run with the `-threaded`,`-O2` options as well as `+RTS -N1` to `-N8` on an 8-core machine.

## References

Coenraad Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph, 1973. URL `https://dl.acm.org/doi/10.1145/362342.362367`.

Stephen A. Cook. The complexity of theorem-proving procedures, 1971. URL https://dl.acm.org/doi/10.1145/800157.805047.

Richard M. Karp. Reducibility among combinatorial problems, 1972. URL https://web.archive.org/web/20110629023717/http://www.cs.berkeley.edu/~luca/cs172/karp.pdf.