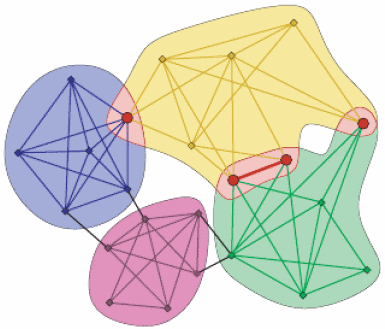
1. **¿Qué es un grafo, sus elementos y propiedades?**

Según el artículo científico un grafo tiene la forma G = (V, E), donde V es un conjunto de vértices y E es el conjunto de aristas en G. El vecindario de un vértice es el conjunto de vértices adyacentes a v, denotado como N(v).

LA vencidad de un vertice “v” se denota como “N(v), es el conjunto de todos los nodos adjacentes a v

1. **Conocer que existen conjuntos de nodos con propiedades especificas de un grafo como son los “maximal cliques” o los “k-cliques”**

**Un Clique** es un subgrafo completo de G donde cada vértice de ese mismo subgrafo es adyacente a cada uno de los demás vértices de ese mismo subgrafo. En la siguiente imagen podemos apreciar varios cliques.



Un **maximal clique** podemos entenderlo como un clique que no puede ser expandido con un nuevo vértice sin perder su propiedad de Clique. Se resuelve de manera exacta y no aproximada.

Un “k-clique” es un clique que tiene exactamente “k” vértices.

MCE ( Maximal clique Enumeration) Busca enumerar todos los maximal cliques de un grafo dato.

1. **Entender cómo se calculan los “maximal cliques” a partir de algoritmos de Bron-Kerbosh con y sin pivotes.**

Texto, Carta

El contenido generado por IA puede ser incorrecto.

* R = Nodos que ya están en el clique actual
* P = Nodos que podrían añadirse al clique (porque son vecinos de todos en R)
* X = Nodos que ya fueron parte de otros cliques que incluían R, se excluyen para evitar duplicados.

Imagen que contiene interior, diversos, tabla, pantalla

El contenido generado por IA puede ser incorrecto.

1. Un reloj con números romanos

   El contenido generado por IA puede ser incorrecto.**Saber enumerar un grafo según su "degeneracy ordering" con el orden alfabético como criterio de desempate.**

Es una técnica que reordena los vértices del grafo para minimizar el número máximo de vecinos de cualquier vértice que aparecen después de él en el orden. Este número máximos se conoce como degeneracy (d) del grafo.

El propósito es reducir el tamaño del conjunto de candidatos P en las primeras etapas del algoritmo, lo que a su vez reduce el tamaño de los subárboles de búsqueda y el coste de las operaciones con conjuntos

Para poder hacer la enumeración vamos vértice a vértice del grafo contando sus vecinos, y ordenamos los vértices de menor grado a mayor grado, desempatando por orden alfabético.

1. **Entender como impacta en la complejidad temporal y de uso de memoria para el problema de la enumeración de “maximal cliques” optar por estrategias BFS y DFS vistas en clase.**

La mayoría de los trabajos anteriores usan búsqueda en amplitud porque se paraleliza fácil, cada nodo de un mismo nivel puede ser tratado de manera paralela. Pero no escala bien en grafos grandes, debido a medida que bajamos de niveles va creciendo de manera sustancial el número de nodos que están en un mismo nivel, por ende, aumenta el uso de memoria.

Otra opción es delegar en la CPU la parte pesada, que es la búsqueda en profundidad y en la GPU se delegan operaciones o funciones de búsquedas más sencillas. Esto deriva en que se necesita una comunicación muy estrecha entre CPU y GPU y esto hace que el coste sea mayor.

1. **Saber calcular los parámetros Δ y d para proporcionar la asíntota del uso de memoria de los conjuntos P y del algoritmo con y sin “degeneracy ordering”.**

**Parámetro Δ (Grado Máximo)**

* **Definición**: El artículo define **Δ** como el **grado máximo del grafo**.
* **Cálculo**: Para calcular Δ en un grafo, debes seguir estos pasos:
  1. Para cada vértice del grafo, cuenta cuántas aristas (conexiones) tiene. Este número es el "grado" de ese vértice.
  2. Compara los grados de todos los vértices.
  3. El valor de Δ es simplemente el grado más alto que encuentres.

Sin degenerancy order:

* El tamaño tanto de P como de X es **.**
* **El tamaño de .**
* **Subgrafo inducido .**

Con degenerancy order:

* El tamaño de P se convierte en
* El tamaño de X se mantiene en
* **Subgrafo inducido .**

1. **Sabes explicar porqué es necesario un workers list vs un work list y como se aplicaría en el segundo nivel de paralelización de árboles en un ejemplo como el de la publicación.**

usar una *work list* tradicional para el problema de la Enumeración de Cliques Máximos (MCE) en GPUs es ineficiente por dos razones principales:

* **Gran tamaño de los datos de la subtarea:** Para definir una subtarea en el algoritmo Bron-Kerbosch, es necesario almacenar mucha información: los conjuntos de vértices R (clique actual), P (posibles candidatos) y X (vértices ya excluidos), además del nivel actual en el árbol de búsqueda y una referencia al subgrafo inducido correspondiente. Guardar toda esta información para múltiples subtareas en una lista consumiría una cantidad excesiva de la ya limitada memoria de la GPU.
* **Alto coste de comunicación:** Añadir y retirar datos tan grandes de la lista de trabajo genera una sobrecarga computacional significativa. Si un bloque de hilos añade una subtarea a la lista, pero no hay ningún bloque ocioso para tomarla de inmediato, ese esfuerzo se desperdicia.

La **worker list** soluciona estos problemas. En lugar de mover grandes volúmenes de datos a una lista central, solo se almacena el identificador del bloque de hilos ocioso. La transferencia de los datos de la subtarea ocurre directamente desde el bloque "donante" (con mucho trabajo) al bloque "receptor" (ocioso), y solo cuando es necesario. Esto minimiza tanto el uso de memoria como la sobrecarga.

1. Conclusiones