Viendo `MaxClique` mediante Simulated Annealing

December 15, 2023

Integrantes: Vicente Sandoval, Matías Jara
MA4402

Temas de la presentación

Preliminares - Definición de Clique

Todos los grafos que se trabajaran serán finitos.
Un grafo G se dirá completo ssi posee todas las aristas posibles.

Se dirá que H, subgrafo de G es clique de tamaño k ssi H es un grafo completo de k vértices, en tal caso también se dice que G posee un clique de tamaño k.

El número de clique de un grafo G, $\omega(G)$ se define como el $k \in \mathbb{N}$ más grande tal que G tiene un clique de tamaño k.

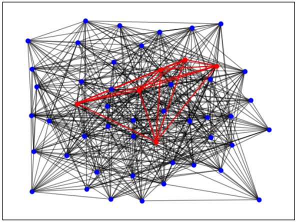


Figure: Clique (Rojo) de tamaño 6

Preliminares - Definición de los Problemas

- Se define el problema de decisión de k-$\textsc{Clique}$, $k \in \mathbb{N}$, como el problema de decidir si existe un k-clique.

- Se define el problema de $\textsc{MaxClique}$, o alternativamente, el problema de optimización de $\textsc{Clique}$, como el problema de encontrar el mayor $k \in \mathbb{N}$ tal que G tenga un k-clique.
  En otras palabras, determinar $\omega(G)$.

- Se define el problema de conteo de $\#$-$\textsc{Clique}$, como el problema de determinar n, donde n es el numero de cliques maximales distintos que contiene G.

Preliminares - Dificultad de los Problemas

- El problema de decisión puede ser resuelto con fuerza bruta bajo un tiempo polinomial. Notese que es importante que el k es parte de la definición del problema.
- El problema de maximización `MaxClique` es un conocido problema NP-completo, hasta ahora no se conocen algoritmos a tiempo polinomial que lo resuelvan.
- El problema de conteo es mucho más difícil que `MaxClique`

Objetivos

- Aplicar Simulated Annealing para resolver el problema de decisión `k-Clique`
  - Estudiar su efectividad
  - Estudiar distintos esquemas de temperatura
  - Identificar grafos donde el algoritmo desempaña bien/mal
- Desarrollar/Extender alguna heurística para el problema de `MaxClique`

k-Clique

k-Clique, Simulated Annealing - Definición

Se considera que los vértices del grafo están enumerados del 0 hasta $n-1$, donde $|V(G)| = n$, sea $\sigma \in S_n$ (El grupo simétrico de tamaño n), se interpretara σ como una permutación de los vértices del grafo, donde los primeros k vértices representan los vertices candidatos a k-clique.

Se define la función F que dada una permutación σ, cuenta la cantidad de aristas en el grafo complemento entre los primeros k nodos de σ, notese que existe un clique en los primeros k vértices ssi aquella cantidad de aristas es 0 (F $=$ 0), la función se escribe como:

$$F(\sigma) = \sum_{i=0}^{k-2} \sum_{j=i+1}^{k-1} (1 - A_{\sigma(i)\sigma(j)})$$

Donde A es la matriz de adyacencia del grafo.

Así, el problema se reduce a encontrar el mínimo de F.

k-Clique, Simulated Annealing - Vecinos

Se implementa el algoritmo de Simulated Annealing descrito en [1] Dado $\sigma \in S_n$, se dirá que $\tau \in S_n$ es vecino de σ ssi τ se obtiene a partir de σ transponiendo uno de sus primeros k elementos con uno de sus últimos $n - k$ elementos, notese que esto corresponde a efectivamente modificar el candidato a clique.

Contando, se tendrá que cada permutación tendrá $k(n - k)$ vecinos, por ende, el grafo de configuraciones asociado sera regular. Se escoge como configuración inicial aquella permutación de nodos donde los nodos estan ordenado según su grado.

k-Clique, Simulated Annealing - Modificaciones

Se introducen modificaciones al algoritmo para promover su desempeño.
Al momento de elegir vecino, se elige de manera uniforme, pero se testea lo siguiente:

- Si agregar el nuevo nodo (y borrar el anterior) aumenta la cantidad de aristas, se acepta y se procede con el Simulated Annealing.
- Si no, se elige otro vecino, si esto vuelve a pasar más de $8n$ veces, se acepta el vecino de todas formas y se procede con el Simulated Annealing.

Esta heurística corresponde a un criterio de búsqueda local, el cual consiste en elegir entre los vecinos, aquel que mejore el valor objetivo, pero dado lo grande que puede ser el grafo de configuraciones, es fácil estancarse en un mínimo local, por ende la segunda condición.

k-Clique, Probando el algoritmo I

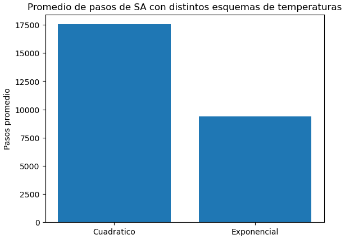
Se testea el algoritmo con grafos aleatorios y grafos de la database de `DIMACS` [3] los cuales tienen número de clique conocido.

G	$\|V\|$	$\|E\|$	ρ	$\omega(G)$	n_{trials}	$\%_{exito}$
C125.9	125	6963	0.898	34	15	$86.\bar{6}$
keller4	171	9435	0.649	11	15	$86.\bar{6}$
p_hat300-1	300	10933	0.243	8	15	20
gen400_p0.9_65	400	71820	0.9	65	5	0
broock400_2	400	59786	0.749	29	5	0

Donde ρ es densidad de aristas.

k-Clique, Probando el algoritmo II

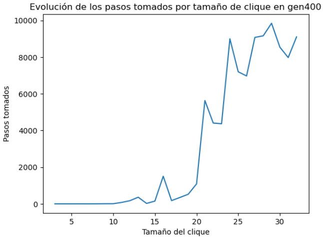
Se genera un grafo aleatorio pequeño ($|V| = 50$) con probabilidad de arista $p = 0.5$ para ejecutar 100 veces el algoritmo, con el número de clique del grafo, con esquema de temperatura cuadrático y exponencial.

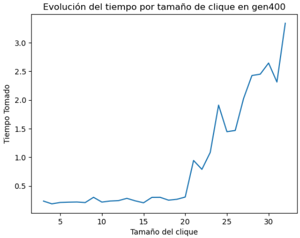


Por ende en adelante se elige exponencial.

k-Clique, Probando el algoritmo III

Se escoge gen400_p0.9_65 ya que es un grafo grande con número de clique alto (65), se ejecuta varias veces el algoritmo, cada vez aumentando el tamaño del clique buscado, y se detiene una vez que el algoritmo es incapaz de encontrar el clique al ejecutar el algoritmo 5 veces. El algoritmo se detiene en un clique de tamaño 33.





k-Clique, Probando el algoritmo IV

Se generan aleatoriamente dos grafos grandes ($|V| = 750$), uno con densidad de aristas ≥ 0.65 y otro con ≤ 0.4, se aproxima el clique máximo que tienen estos grafos y se ejecuta el algoritmo varias veces hasta que, en algún momento, finalice correctamente.

En el caso del grafo con densidad baja, el algoritmo se tuvo que ejecutar a lo más 4 veces para que termine de manera exitosa.

En el caso del grafo con densidad alta, el algoritmo se ejecuto 25 veces, y ninguna termino de manera exitosa.

`MaxClique`

MaxClique - Construyendo el grafo de configuraciones

Motivados por lo anterior, se considera una configuración como un par (σ, k), donde σ ejerce el mismo rol que antes y k representa el tamaño del clique en cuestión.

Se construye el grafo de configuraciones que consiste en tomar todos los grafos de configuraciones anteriores (había uno por cada k), etiquetándolos con su respectivo k para diferenciarlos.

Se consideran dos grafos de configuraciones, el primero (Naive) consiste en añadir las aristas de transición de estilo $(\sigma, k) \to (\sigma, k+1), (\sigma, k) \to (\sigma, k-1)$, donde k es tomado con modulo n, el segundo (Salto) consiste en las aristas anteriores más las aristas tipo $(\sigma, k) \to (\sigma, n-k)$.

MaxClique - Función a minimizar

Sea F_k la función objetivo del problema `k-Clique`, la idea es que si el par (σ, k) efectivamente representa un clique de tamaño k, entonces $F_k(\sigma) = 0$, pero si desea el k más grande que cumpla esto, por ende se añade una penalización a la función objetivo que favorezca k más grandes, así, la función objetivo queda como

$$F(\sigma, k) = \alpha F_k(\sigma) + p(k)$$

Donde $\alpha \in \mathbb{R}$ es una sensibilidad a elegir, y p es la función de penalización, se elige $\alpha = 1, p(k) = k^{-2}$.

MaxClique - Detección de cliques intermedios

En cada paso del Simulated Annealing, se calculara el valor de la función objetivo, en particular, se calcula el valor de $F_k(\sigma)$, si este llega a ser igual a 0, entonces el algoritmo ha encontrado un clique de tamaño k, teniendo esto en cuenta, se programa el algoritmo para que vaya almacenando los cliques que va encontrando.

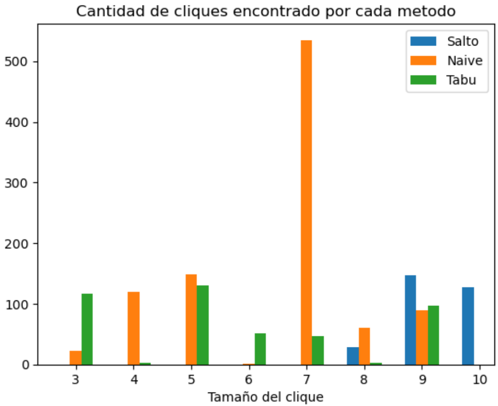
No es esperable que esto de una solución a el problema de conteo debido a la cantidad gigante de cliques que pueden existir, aun así, puede ser interesante ver cuantos cliques recolecta según cada método.

`MaxClique` - Prohibición tipo tabú de corto plazo

Al inspeccionar los algoritmos anteriores, da la impresión de que el algoritmo puede recorrer muchas veces el mismo candidato a clique, pero recorriendo nodos del grafo de configuración distintos, por ende se implementa una tercera versión, a partir del `Naive`, que consiste en adicionarle una corta memoria tabú (de tamaño máximo 100), en que guarda los últimos 100 nodos visitados y si algún nodo esta en memoria, no lo vuelve a visitar.

Para implementar esto de manera correcta, hay que condensar todos los nodos del grafo de configuración que representan un mismo candidato a clique en un solo nodo, esto se logra rompiendo todas las posibles simetrías que las configuraciones pueden tener, si se tiene (σ, k), se ordena de mayor a menor los primeros k elementos de σ y de mayor a menor los últimos $n - k$ elementos de σ.

MaxClique - Desempeño en C125.9



- El tiempo tomado por el algoritmo con salto es: 25.702 s
- El tiempo tomado por el algoritmo naive es: 49.102 s
- El tiempo tomado por el algoritmo tabu es : 67.329 s

Conclusiones

- El algoritmo original (`k-Clique`) funciona bien para grafos pequeños y/o números de clique pequeños, también parece funcionar bien en grafos con densidad de arista baja
- El algoritmo no funciona de manera deseable para grafos grandes o números de clique grande, por ende aplicarlo para `MaxClique` no es factible
- Se exploran heuristicas alternativas, las cuales no son muy exitosas en encontrar el clique maximo.

Referencias

Xiutang Geng, Jin Xu, Jianhua Xiao, Linqiang Pan
A simple simulated annealing algorithm for the maximum clique problem
Information Sciences, Volume 177, Issue 22, 15 November 2007, Pages 5064-5071

Fred Glover
Tabu Search: A Tutorial. Interfaces
Interfaces,Vol. 20, No. 4, 1990.

Second DIMACS Implementation Challenge
October 11-13, 1993, DIMACS benchmark set.