MÍNIM CONJUNT DOMINADOR D'INFLUENCIA POSITIVA

Pol Casacuberta
Joan Morilla
Miquel Umbert

Part 1

Apartat a)

1- Descripció del problema:

Donat un graf $G = (V, E)$ i un conjunt $D \subseteq V$es demana comprovar si D és conjunt dominador d'influència positiva en G, i en cas de ser-ho, comprovar si és minimal.
Aquest problema es divideix en dues parts, primer, comprovar que D sigui conjunt dominador d'influència positiva en G, per a veure això hem de comprovar que almenys la meitat dels veïns de cadascun dels vèrtexs que pertanyen a G, també pertanyen al conjunt D.
Per a la segona part del problema se'ns demana veure que D en cas de ser conjunt dominador d'influència positiva en G no hi hagi subconjunt H on $H \subset D$ i també sigui conjunt dominador d'influència positiva en G.

2- Explicació de l'algoritme i cost:

Algoritme 1:
Un cop vist què és el que ens demanen, procedim a explicar la nostra proposta de l'algoritme dissenyat per a la primera part de la pregunta.
La idea és, iterar sobre tots els vèrtexs que pertanyen a G i per cada vèrtex iterar sobre els seus veïns, i per cada un d'aquests veïns buscar-lo en el set D, si el trobem sumem 1 a un comptador que ens conta el nombre de veïns dins del conjunt D. Quan acabem d'iterar sobre els veïns d'un vèrtex, comprovem que el nombre de veïns dins el conjunt D és major o igual que el ceiling de veïns partit entre dos en cas que no ho sigui, l'algoritme retorna false.

Cost:
Iterar sobre cada vèrtex i sobre els veïns O(V^2) buscar dins d'un set tenim cost aproximat de log(n) on n és el nombre d'elements en el set D i sumar un comptador és O(1) per tant, el cost pitjor és de O(V² *log(n)).

Algoritme 2:
Per a aquesta segona part de la pregunta començarem iterant sobre els vèrtexs v_i del conjunt D i sobre els veïns d'aquests vèrtexs v_j, sobre aquests iterarem sobre els seus veïns v_k per a comprovar que si traiem 1 vèrtex els veïns del vèrtex v_i, segueixen tenint més o igual de veïns v_k que el ceiling de veïns partit entre dos. En cas que no es compleixi la condició anterior considerarem el següent vèrtex v_i sense acabar d'iterar per la resta de veïns v_j. Si, per altra banda, es compleix la condició per a tots els veïns de v_i, llavors hem trobat un vèrtex que podem treure, i se seguirà complint la condició de ser conjunt dominador d'influència positiva en G.

Cost: Iterar sobre els vèrtexs i els seus veïns té cost O(D^2) on D és el nombre de vèrtexs en el conjunt D. Iterar sobre els veïns dels veïns té un cost màxim O(D²*V) on V és el nombre de vèrtexs en G. Finalment fem una cerca en el conjunt D dins del set D per cada veí del veí, sumant un cost O(D²*V*log(D)).

3- Justificació de les estructures de dades

Hem fet servir un set per al conjunt D, ja que ens permet fer la cerca en temps logarítmic i no pas en cost lineal com tindríem amb un vector.

Apartat b)

1- Descripció del problema:

En aquest apartat ens demanen dissenyar i implementar un algoritme voraç determinista que trobi un conjunt dominador d'influència positiva donat un graf $G = (V, E)$.

2- Explicació de l'algoritme i cost

Com ens demanen un subconjunt de vèrtexs $S \subset V$ que sigui dominador positiu, S haurà de tindre les següents característiques. En primer lloc, el nombre d'elements del conjunt S haurà de ser com a mínim més gran que la meitat del nombre d'elements de V, així assegurem que serà positiu. En segon lloc, els vèrtexs corresponents al conjunt S, han de cobrir totes les arestes al graf G.

Algoritme: Per començar tindrem el graf com una llista adjacència on V[i] contindrà els veïns del vèrtex *i.* Per aconseguir un conjunt dominador positiu començarem ordenant el vector de conjunts V per la mida de cada conjunt (reanomenem el graf pel grau de cada vèrtex). Un cop ordenat la idea d'aquest algorisme és anar agafant els vèrtexs amb més veïns fins a haver-ne cobert totes les arestes del graf. Però com ha de ser positiu, continuarem afegint al conjunt S els vèrtexs en ordre decreixent de grau fins que la mida del conjunt sigui més gran que la meitat del conjunt V i totes les arestes siguin cobertes.

Aquest algoritme ens retornarà un conjunt de vèrtex que compleix que sigui dominant i positiu, però la quantitat de vèrtex que selecciona és igual o superior al conjunt dominant i positiu mínim.

Cost: Com que la idea és anar escollint els vèrtexs d'un en un de manera contínua fins que els criteris s'han complert, el cost de seleccionar els vèrtexs és lineal O($|V|$). Per altra banda, per dur a terme aquest algoritme és necessari ordenar els vèrtexs, per tant, depenent de l'algoritme d'ordenació podem aconseguir un cost temporal per aquesta part de l'ordre de O($n\ log^2(n)$). Les dues parts combinades ens donen una complexitat temporal de l'algoritme de O($n\ log^2(n)$).

3- Justificació estructura de dades.

Hem utilitzat un set per anar seleccionant els vèrtexs del conjunt S, ja que ens permet fer cerca dels vèrtexs ja presents en el conjunt en temps logarítmic, a més de tindre'ls ordenats per una major llegibilitat

Apartat c)

1- Descripció del problema:

En aquest apartat ens demanen que a partir d'un graf $G = (V, E)$, hem de trobar un conjunt dominador d'influència positiva a G mitjançant un algoritme de cerca local, i aquest podem escollir que sigui determinista (Hill Climbing), o estocastic (Simulated Annealing) com és el nostre cas.

2- Explicació de l'algoritme i cost

Al principi vam voler fer aquest apartat amb un vector de booleans, per tant, si vec[i]=true, significava que el node i estaria inclòs en la nostra solució, per tant, voldríem minimitzar el nombre de posicions a true del vector. Però vam veure que les operacions amb vectors tardaven molt a executar-se, per això vam optar per tenir 2 sets, un set contendria els nodes que estarien en la nostra solució, i l'altre set contendria la resta de nodes, d'aquesta forma l'execució del nostre programa és molt més ràpida.

Comencem amb el set de solucions complet, amb tots els nodes dins del conjunt dominador; així ens assegurem de què estem en una solució vàlida. Un cop omplert el vector, anirem iterant damunt la nostra solució, on bàsicament només ens faran falta les opcions de llevar un vèrtex al conjunt dominador actual, ja que la solució inicial donada ens permet explorar tot l'espai de solucions que tenim, i en cas que no es pugui llevar cap vèrtex, es contemplaria l'opció d'afegir un vèrtex al conjunt dominador actual, en aquest cas no estem complint l'objectiu de minimitzar el nombre de vèrtexs seleccionats, però ens permet no quedar-nos estancats en un mínim local, però amb el temps, l'algoritme de Simmulated Annealing deixarà d'inserir vèrtexs en la possible solució, fent que només creem solucions òptimes.

Així que primer de tot mirem quins són els vèrtexs que podem llevar del conjunt actual sense que aquest deixi de ser un MPIDS, per tant ens assegurem de no sortir de l'espai de solucions.

De tots aquests vèrtexs, n'elegim un a l'atzar per llevar-lo del conjunt, i, per tant, obtenim un successor. Anirem iterant d'aquesta forma fins que arribem al nombre d'iteracions donat o no podem llevar vèrtexs del conjunt, en el segon cas, haurem d'afegir algun vèrtex nou.

El cost principal de l'algorisme és escollir quins vèrtexs es poden llevar del conjunt i que aquest segueixi sent un MPIDS, per tant, el cost, seria O($n(n)$) -> O(n^2)

3- Justificació estructura de dades.

Crèiem que la forma més òptima i fàcil per aplicar aquest apartat és amb vector de booleans, així sabem en tot moment quins vèrtexs estan actualment escollits com a possible solució. Però pel fet que seleccionar els elements d'un vector és un procés molt més lent que en un set, hem optat per fer 2 sets, un que contendrà els nodes del conjunt dominant i un altre que contendrà la resta.

Els experiments han estat executats tots amb un màxim de 1000 iteracions per a l'algorisme de Simulated Annealing.

Part 2

Apartat e)

1- Descripció del problema:

Implementar el model ILP del problema MPIDS, on busquem minimitzar la suma dels vèrtexs que pertanyen al conjunt dominador d'influència positiva, però mantenint la propietat dels conjunts dominadors d'influència positiva és a dir estem subjectes a què la suma dels veïns de cada vèrtex que pertanyen al conjunt dominador d'influència positiva sigui major o igual al ceiling de veïns partit entre dos.

2- Explicació de l'algoritme i cost:

Gràcies a la llibreria Cplex podem implementar de forma directa el model ILP. Primer iniciem un array de variables binàries amb el mínim valor (0) que podem assignar i el valor màxim (1) que ens serviran per representar si els vèrtexs de G pertanyen o no al conjunt D. Iniciem també l'expressió que volem minimitzar i iterem per a cada vèrtex de G i sumem a l'expressió que volem minimitzar el valor de cada variable de l'array.
Un cop hem definit els paràmetres a minimitzar hem d'afegir quines restriccions volem que s'acompleixin. Iterem sobre tots els vèrtexs del graf G i afegirem una restricció en forma d'expressió per cada un d'aquests vèrtexs, iterem per sobre dels veïns de cada vèrtex i el sumem a l'expressió, finalment afegirem la restricció on l'expressió que hem calculat ha de ser major o igual al ceiling de veïns partit entre dos.

El cost de calcular les restriccions és de $O(V^2)$, ja que per cada vèrtex iterem sobre els seus veïns i cada vèrtex pot estar connectat com a molt el nombre de vèrtexs menys un. El cost de calcular l'expressió a minimitzar és lineal en el nombre de vèrtexs O(V), per tant, el cost total és de $O(V^2)$.

3. Experiments:

Per a provar l'efectivitat de la llibreria Cplex hem executat l'algoritme per als diversos jocs de prova, on ha trobat per a algunes instàncies com la de graph_football o la de graph_jazz la millor solució, en canvi, per a altres l'algoritme no ha arribat a trobar la millor solució en el temps donat, però sí que troba molt bones aproximacions.

Bibliografia

[1] Salim Bouamama and Christian Blum. An improved greedy heuristic for the minimum positive influence dominating set problem in social networks. Algorithms, 14(3):79, 2021.

[2] Cai, S.; Hou, W.; Wang, Y.; Luo, C.; Lin, Q. Two-goal Local Search and Inference Rules for Minimum Dominating Set. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, Yokohama, Japan, 11–17 July 2020; pp. 1467–1473.

[3] Yiyuan Wang, Shaowei Cai, Jiejiang Chen, and Minghao Yin. A fast local search algorithm for minimum weight dominating set problem on massive graphs. In Proceedings of IJCAI 2018, pages 1514–1522, 2018.]

[4] Yi Fan, Yongxuan Lai, Chengqian Li, Nan Li, Zongjie Ma, Jun Zhou, Longin Jan Latecki, and Kaile Su. Efficient local search for minimum dominating sets in large graphs. In Proceedings of DASFAA 2019, pages 211–228, 2019.

[5] Lin, G.; Guan, J.; Feng, H. An ILP based memetic algorithm for finding minimum positive influence dominating sets in social networks. Phys. Stat. Mech. Appl. 2018, 500, 199–209.

[6] IBM ILOG CPLEX Optimization Studio :
https://www.ibm.com/docs/en/icos/20.1.0?topic=cplex