This is a post-peer-review, pre-copyedit version of a paper published in Arai K, Kapoor S & Bhatia R (eds.) *Intelligent Computing. SAI 2020.* Advances in Intelligent Systems and Computing, 1228. Cham, Switzerland: Springer International Publishing, pp. 122-136. The final authenticated version is available online at: https://doi.org/10.1007/978-3-030-52249-0_9

Finding the Maximal Independent Sets of a Graph Including the Maximum Using a Multivariable Continuous Polynomial Objective Optimization Formulation

Maher Heal and Jingpeng Li

Department of Computing Science and Mathematics,
University of Stirling, Stirling, UK
{maher.heal, jli}@cs.stir.ac.uk

Abstract. We propose a multivariable continuous polynomial optimization formulation to find arbitrary maximal independent sets of any size for any graph. A local optima of the optimization problem yields a maximal independent set, while the global optima yields a maximum independent set. The solution is two phases. The first phase is listing all the maximal cliques of the graph and the second phase is solving the optimization problem. We believe that our algorithm is efficient for sparse graphs, for which there exist fast algorithms to list their maximal cliques. Our algorithm was tested on some of the DIMACS maximum clique benchmarks and produced results efficiently. In some cases our algorithm outperforms other algorithms, such as cliquer.

Keywords: Independent set, continuous optimization, MATLAB, maximal cliques, sparse graphs.

1 Introduction

The maximum independent set problem and the maximal independent set problem of a certain size and the related problems of maximum/maximal cliques are important problems in combinatorial optimization. They have many applications in diverse range of domains such as computer vision/pattern recognition, information/coding theory, molecular biology and scheduling. As the problems are NP-hard, it is unlikely there will be an ultimate solution to the problems unless P=NP. However, many algorithms and heuristics were proposed to solve the problems for certain graphs [1–4]. In this paper we confine ourselves with quadratic and continuous programming formulations to find the maximum(maximal) independent set(s). We propose new quadratic and continuous polynomial formulations to find these graph invariants. Our formulations are most suitable for sparse graphs, since we need to list the maximal cliques of the graph first to find the maximum(maximal) independent set(s). Second the nonlinear optimization solvers - we used matlab - are more efficient for sparse graphs. The paper is organized as follows: section 1 is the introduction, section 2

is the literature review of mainly continuous optimization algorithms to solve the independent set problem, section 3.1 explains our formulation using a quadratic objective function, section 3.2 extends that formulation to multivariable polynomial objective function formulation, section 4 gives a second proof to our main result, section 5 gives a geometric third proof to our formulation, section 6 gives examples and extensions to our formulation, section 7 are results obtained by applying our algorithm to DIMACS list of clique benchmarks and finally section 8 is the conclusion.

2 Literature Review

First of all, assume we have a finite graph $G(\mathcal{N}, \mathcal{L})$. $\mathcal{N}$ is the set of N vertices and $\mathcal{L}$ is the set of L edges. Associate with each vertex i a continuous variable θ_i.

N.Z. Shor [5] proved that the binary formulation in Eq. 1 is equivalent to the following quadratic optimization formulation:

$$\begin{aligned} max \quad & \theta_1 + \theta_2 + ... + \theta_N \\ & \theta_i \theta_j = 0, \forall (i,j) \in \mathcal{L} \\ & \theta_i^2 - \theta_i = 0, i = 1, 2, ..., N \end{aligned}$$

and reported good computational results. In [6] Motzkin and Straus found a noteworthy relation between the maximum clique of a graph and the following quadratic programming problem. They proved that the global optima of:

$$\begin{aligned} max \quad & f(x) = \frac{1}{2} \theta^T A_G \theta \\ & e^T \theta = 1 \\ & \theta \geq 0 \end{aligned}$$

is given by

$$\frac{1}{2}(1 - \frac{1}{\omega(G)})$$

where $\omega(G)$ is the clique number of graph G, A_G is the adjacency matrix of the graph and e is an N dimensional vector of all 1s. Harant et al [7, 8] proved the following continuous and quadratic formulations about the independence number of a graph. First the continuous polynomial formulation is given by:

$$\alpha(G) \quad = \quad \max_{0 \leq \theta_i \leq 1, i=1,...,N} F(x) \quad = \quad \max_{0 \leq \theta_i \leq 1, i=1,...,N} \sum_{i=1}^{N} (1 \ - \ \theta_i) \prod_{(i,j) \in \mathcal{L}} \theta_j$$

and the quadratic formulation is given by:

$$\alpha(G) \quad = \quad \max_{0\leq\theta_i\leq 1, i=1,\ldots,N} H(x) \quad = \quad \max_{0\leq\theta_i\leq 1, i=1,\ldots,N} (\sum_{i=1}^{N}\theta_i \; - \; \sum_{(i,j)\in\mathcal{L}} \theta_i\theta_j)$$

where $\alpha(G)$ is the independence number of the graph. It is clear the Harant formulations are optimizations over a unit hypercube. Other formulations reported in the literature over a hyper sphere, such as [9]. They proved that if $\bar{\theta}$ is a solution to the following optimization problem:

$$V(k) = \min \frac{1}{2}\theta^T A_G \theta + (\sum_{i=1}^{N}\theta_i - 1)^2$$

subject to:

$$\sum_{i=1}^{N}\theta_i^2 \leq \frac{1}{k}$$

$$\theta \geq 0$$

then $V(k) = 0$ iff there exists an independent set I in G such that $|I| \geq k$.

3 The Formulation

We state here the main result that sets out the main body of our algorithm. First, we prove the result for a quadratic programming formulation, then we extend it to a multivariable continuous polynomial programming formulation. Again as stated in section 2, assume we have a finite graph $G(\mathcal{N}, \mathcal{L})$. $\mathcal{N}$ is the set of N vertices and $\mathcal{L}$ is the set of L edges. Associate with each vertex i a continuous variable θ_i.

Before we state our formulation, it is good to recall a binary programming formulation to find the maximum independent set of any graph. It is well known that the solution of the following binary programming optimization problem yields the maximum independent set:

$$\begin{aligned} max \quad & \theta_1 + \theta_2 + ... + \theta_N \\ \sum \theta_i \leq 1 \quad & \text{at each maximal clique} \\ \theta_i \in \{0,1\} & i = 1, 2, ..., N \end{aligned} \tag{1}$$

3.1 Quadratic Formulation

A global solution of the quadratic optimization problem

$$\begin{aligned} max \quad & \Psi \\ \sum \theta_i \leq 1 \quad & \text{at each maximal clique} \end{aligned}$$

$$0 \leq \theta_i \leq 1 \tag{2}$$

where $\Psi = \theta_1^2 + \theta_2^2 + ... + \theta_N^2$ is the independence number of the graph and the solution vector is a binary vector of 1s and 0s. Moreover, the vertices that form a maximum independent set, have their θ variable equals 1 and all other θs equal to 0. In addition, a local maxima of Ψ under the same conditions is a binary vector such that the θ variable equals to 1 for vertices that form a maximal independent set and 0 for the rest.

proof:

We will prove that a local maxima is a maximal independent set with a solution vector that has θ_i=1 $\forall i \in$ the maximal independent set and θ_i=0 $\forall i \notin$ the maximal independent set. Let $i = 1, 2, ..., p$ be a maximal independent set. We need to prove $\theta_i = 1$ for $i = 1, 2, ..., p$ and $\theta_i = 0$ for $i = p+1, ..., N$ is a local maxima.

Recall the definition of a local maxima of a multivariable function $f(x_1, x_2, ..., x_n)$. $(x_1^*, x_2^*, ..., x_n^*)$ is a local maxima if $\exists \epsilon > 0$ such that

$$f(x_1, x_2, ..., x_n) \leq f(x_1^*, x_2^*, ..., x_n^*) \forall$$

$$|x_1 - x_1^*| \leq \epsilon, |x_2 - x_2^*| \leq \epsilon, ..., |x_n - x_n^*| \leq \epsilon$$

As each $0 \leq \theta_i \leq 1$, we need to prove that $\exists \epsilon$ such that for $0 < \epsilon \leq 1$ and at

$$1 - \epsilon \leq \theta_i \leq 1, i = 1, ..., p$$

$$0 \leq \theta_i \leq \epsilon, i = p + 1, ..., N$$

Ψ is less than that Ψ at $\theta_i = 1, i = 1, ..., p$ and $\theta_i = 0, i = p + 1, ..., N$. However, since $\{1, ..., p\}$ is a maximal independent set, then each $j \in \{p + 1, ..., N\}$ must be connected to at least one of $\{1, ..., p\}$ and each maximal clique contains one and only one of $\{1, ..., p\}$. Given that

$$\sum \theta_i \leq 1 \quad \text{at each maximal clique}$$

and taking $0 \leq \theta_i \leq 1 - \Delta, i = 1, ..., p$, $0 < \Delta \leq 1$, we must have $0 \leq \theta_i \leq \Delta, i = p + 1, ..., N$. Now

$$\Psi = \theta_1^2 + ... + \theta_N^2 \leq \underbrace{(1 - \Delta)^2 + ... + (1 - \Delta)^2}_{\text{p times}} + \underbrace{\Delta^2 + ... + \Delta^2}_{\text{N-p times}}$$

Three cases are considered now:

case I: $p = N - p$

$$\Psi \leq \underbrace{(1 - \Delta)^2 + \Delta^2 + ... + (1 - \Delta)^2 + \Delta^2}_{\text{p times}}$$

Since $(1-\Delta)^2+\Delta^2 \leq 1$ because $a^2+b^2 \leq (a+b)^2$ for positive a and b, we have $\Psi \leq p$. We can take ϵ any value between 0 and 1 inclusive.

case II: $N-p<p$

$$\Psi \leq \underbrace{(1-\Delta)^2+\Delta^2+...+(1-\Delta)^2+\Delta^2}_{\text{N-p times}}+\underbrace{(1-\Delta)^2+...+(1-\Delta)^2}_{\text{2p-N times}} \leq N-p+2p-N \leq p$$

again we can take ϵ any value between 0 and 1 inclusive.

case III: $N-p>p$

$$\Psi \leq \underbrace{(1-\Delta)^2+\Delta^2+...+(1-\Delta)^+\Delta^2}_{\text{p-1 times}}+(1-\Delta)^2+(N-2p+1)\Delta^2$$

for $\Delta \leq \frac{2}{N-2p+2}$we have $\Psi \leq p$. Taking the limit as $\Delta \to 0$, we have $\Psi \to p$, $\theta_i \to 1, i=1,...,p$ and $\theta i \to 0, i=p+1,...,N$. Recalling the definition of the limit, for each $\epsilon>0$ $\exists\delta$ such that $|\theta_i-1| \leq \epsilon, i=1,...,p$ and $|\theta_i-0| \leq \epsilon, i=p+1,...,N$ for all $|\Delta-0| \leq \delta$, or $1-\epsilon \leq \theta_i \leq 1, i=1,...,p$ and $0 \leq \theta_i \leq \epsilon, i=p+1,...,N$ for some δ such that $0 \leq \Delta \leq \delta$. Take the ϵ that corresponds to $\delta=\frac{2}{N-2p+2}$, so accordingly we found the ϵ.

Clearly the global maxima is a maximum independent set.

3.2 Extension to Multivariable Polynomial Formulation

We show that if

$$\Psi=\theta_1^r+...+\theta_N^r$$

$r>1$ the result proved in 3.1 is correct. This surely includes Ψ is a polynomial with $r \geq 2$. Following the same logic of proof in 3.1 case I and II are correct since $x^r+(1-x)^r \leq x+(1-x) \leq 1$ and $(1-x)^r \leq 1$ for $0 \leq x \leq 1$ and $r>1$. For case III

$$\Psi=\underbrace{(1-\Delta)^r+\Delta^r+...+(1-\Delta)^r+\Delta^r}_{\text{p-1 times}}+(1-\Delta)^r+Q\Delta^r$$

$$Q=N-2p+1$$

the function $f(x)=(1-x)^r+Qx^r$ is convex with one minimum at $x_0=\frac{1}{1+Q^{\frac{1}{r-1}}}$. This can be shown using calculus. At a local minima

$$\frac{df(x)}{dx}=0$$

and

$$\frac{d^2f(x)}{dx^2}>0$$

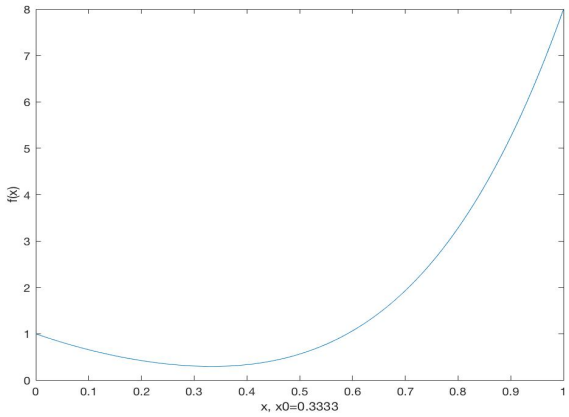
$$\frac{df(x)}{dx} = -r(1-x)^{r-1} + rQx^{r-1} = 0$$

solving for x we have $x_0 = \frac{1}{1+Q^{\frac{1}{r-1}}}$. Now

$$\frac{d^2f(x_0)}{dx^2} \;=\; r(r-1)(1-\frac{1}{1+Q^{\frac{1}{r-1}}})^{r-2} + r(r-1)Q(\frac{1}{1+Q^{\frac{1}{r-1}}})^{r-2} \;>\; 0$$

Fig. 1 shows a sample graph of $f(x)$. It is clear for $\Delta < x_0$ $f(x) \leq 1$ and hence $\Psi < p$. Now by using the same logic as the proof in 3.1 as $\Delta \to 0$, we can find the ϵ.

Fig. 1: graph of $f(x), Q = 8$ and $r = 4$. $x_0 = 0.3333$



4 A Second Proof

It has been shown by Jain et al [10] (please refer to that paper to understand his network flow model under interference) that if the maximal independent sets are $I_1, I_2, ..., I_k$ then

$$f_i = \sum_{i \in I_j} \lambda_j$$

$$\lambda_1 + \lambda_2 + ... + \lambda_k = 1$$

$$0 \leq \lambda_j \leq 1, j = 1, 2, ..., k$$

and

$$\sum_{i=1}^{N} f_i = \text{independence number}$$

for a two nodes network. λ_j is the time allocated to independent set I_j. It can be easily seen :

$$Max \quad \Psi$$

$$\sum \theta_i \leq 1 \quad \text{at each maximal clique}$$

$$0 \leq \theta_i \leq 1 \tag{3}$$

given that Ψ is the quadratic or polynomial function as in 3.1 or 3.2 is equivalent to :

$$Max \quad \Psi$$

$$\sum \theta_i \leq 1 \quad \text{at each maximal clique}$$

$$0 \leq \theta_i \leq 1$$

$$\theta_i = \sum_{i \in I_j} \lambda_j \quad 0 \leq \lambda_j \leq 1, j = 1, 2, ..., k. \tag{4}$$

However:

$$\lambda_1 + \lambda_2 + ... + \lambda_k$$

may or may not equal to 1. Now, we prove

$$\lambda_1 = 1, \lambda_2 = \lambda_3 = ... = \lambda_k = 0$$

which is equivalent to

$$\theta_i = 1 \forall i \in I_1 \quad \text{and} \quad \theta_i = 0 \forall i \notin I_1$$

is a local maxima to (3). The same can be proved for I_2,I_3,...,I_k. Let $\lambda_1 \leq 1 - \Delta$. Now each one of the other independent sets $I_2, l_3, ..., l_k$ either contains a vertex that is also a member of I_1 and in that case λ of that independent set is $\leq \Delta$ since $\theta_i \leq 1$; or the 2nd case is that the independent set contains a vertex that is not a member of the independent set I_1. Hence, this vertex must be connected to one of of the of the vertices of independent set I_1. However, since $\sum \theta_i \leq 1$ at each maximal clique we have $\lambda \leq \Delta$ for this independent set. Based on that

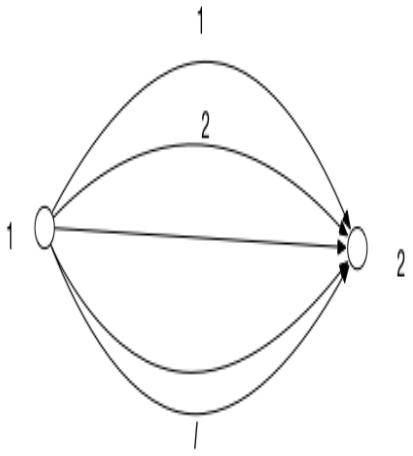
$$\Psi \ = \ \theta_1^2 + \theta_2^2 + ... + \theta_N^2 \ \leq \ \underbrace{(1-\Delta)^2 + (1-\Delta)^2 + ... + (1-\Delta)^2}_{|I_1| \text{ times}} + Z\Delta^2 \tag{5}$$

for some integer Z. The proof now proceeds as in 3.1 for the three cases. Similarly, we can prove it for any power more than 1 in Ψ.

5 A Third Geometric Proof

We will illustrate the proof by considering a four vertices graph, and it is clear to see that can be extended to graphs of arbitrary size. Furthermore the proof suggests an extension of our model to find the maximal weighted independent sets including the maximum weighted independent set. Now, let's us state a connection between the maximum independent sets and the capacity of flow of two nodes network. Consider a two nodes network, see Fig. 2. Node 1 is sending data to node 2 over l links(channels) in one unit of time and the capacity of each link is one data unit per time unit. The links are interfering in such way that the transmission from node 1 to node 2 is successful only when the data is transmitted over non-interfering links. Assuming there is a one-to-one mapping between the links in this two nodes network and the vertices of the graph $G = (\mathcal{N}, \mathcal{L})$, provided that two links in the network are interfering if and only if the corresponding vertices of the graph G are connected. It is not difficult to see the independence number of the graph G is equal to the maximum successful flow from node 1 to node 2. The maximum flow of the network can be carried on any set of links that maps to a maximum independent set of the graph, [10]. We assume $\theta_i, i = 1, 2, ...l$, $l = N$, as the flow in data units/time unit for each link.

Fig. 2: Two nodes networks of l links that conflict according to graph G.



Consider the graph in Fig. 3, it is clear there are two maximum independent sets (1, 2, 3) and (1, 2, 4) and one maximal clique (3, 4). Link (vertex) 3 is interfering(connected) with link (vertex) 4. It is clear the maximum successful flow from node 1 to node 2 is equal to the independence number 3. Now when $\theta_1 = \theta_2 = \theta_3 = 1$ and $\theta_4 = 0$ we have maximum flow from node 1 to node 2, see Fig. 4 (a) ; or it can be attained by splitting the flow of link (vertex) 3 between links (vertices) 3 and 4 since we have the sum of flows is less than or equal to

Fig. 3: Four vertices graph example.

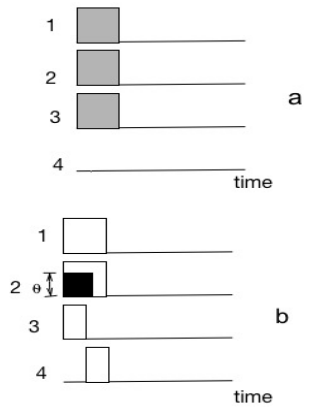
one in each maximal clique, see Fig. 4 (b). It is clear the total areas of the grey squares (each square has a side length of 1 unit) is equal to maximum transmitted data and maximum transmitted data equals to the independence number of the graph. It can be easily seen for any flows of θs we have $\theta_1^2+\theta_2^2+...\theta_l^2$, $l = N$, is less than the independence number of the graph or maximum transmitted data and we attain the maximum when links(vertices) form a maximum independent set and θs equal 1 for maximum independent links(vertices) set and zero otherwise since the area of the inner squares (black square which has a side length of θ) are less than 1, see Fig. 4 (b). It is straightforward to see the logic is valid if we consider a set of links that forms a maximal independent set and not for only a maximum independent set.

Now if we assume links capacities are different such as C_1, C_2 and $...C_l$ for links $1, 2, ...l$ respectively, then it is not difficult to see that the maximum of

$$C_1\theta_1^2 + C_2\theta_2^2 + ... + C_l\theta_l^2$$

is a maximum weighted independent set or a maximal weighted independent set such that the capacities are links weights and depending on if it is a global or local maximum respectively. This should be the objective function in our optimization formulation. As an example if the weights of our four vertices graph in Fig. 3 are 1,2, 3, 4 for vertices 1, 2, 3, 4 respectively, then the maximum of data sent (which is equivalent to maximum weighted independent set) will be 1+2+4=7 or the maximum weighted independent set will be links (vertices) 1,2 and 4.

Fig. 4: Schedule of flows for a network that has the four vertices graph as a conflict graph (a) an independent set carries all the flow (b)link 3 flow is splitted over 2 links, also inner square area is less 1.



6 Examples and Extensions

As an example, we applied our algorithm to Hoffman Singleton graph [11]. After more than 7 hours of computer crunching, WolfRam Mathematica 11.01 FindIndependentVertexSet function didn't converge to a maximum independent set on MacBook Pro, 2.5 GHz Intel Core i5, 8 GB memory. We used Matlab R2017a to code our algorithm. The first phase is finding the maximal cliques of the graph. To that end we used a code from matlab File Exchange for Bron-Kerbosch algorithm to list the maximal cliques [12]. In spite of that this algorithm is not the best known algorithm for large sparse graphs, but it did serve our purpose. There are almost polynomial algorithms to list maximal cliques for sparse graphs reported in literature such as [13]. The matlab code is shown below:

```
function [finalx,finalfval,exitflag,output] = conversionA
    (Adjacency)
% convert the Adjacency matrix into the optimization
    parameters of
% fmincon
%************************************
% LB<=X <=UB
t0=clock;
A=[];
Max_Ind_Set=[];
novertices=size(Adjacency,1);
%************************************
%calculating A and B, LB and UB
t1=clock;
```

```
P=maximalCliques(Adjacency,'v2');
t2=clock;
e1=etime(t2,t1);
disp(e1/60);
c_constraints=size(P, 2);   %number of clique constraints
%************************************
%calculating LB
LB=zeros(novertices,1);
%************************************
%calculating UB
UB=ones(novertices,1);
%************************************
%calculating B
B=ones(c_constraints,1);
%************************************
%theta_1+theta_2+ ... + theta_r_i <=1
A=P';
t3=clock;
x0=rand(novertices,1);
ptmatrix=rand(100,novertices);
tpoints = CustomStartPointSet(ptmatrix);
options = optimoptions('fmincon','Algorithm','sqp','
    SpecifyObjectiveGradient',true,'display','off');
problem=createOptimProblem('fmincon','objective',
    @throughput,'x0',x0,'Aineq',A,'bineq',B,'options',
    options,'lb',LB,'ub',UB);
ms = MultiStart('UseParallel',true,'XTolerance',1);
[x,fval,exitflag,output]= run(ms,problem,tpoints);
finalfval=-fval;
finalx=x;
t4=clock;
e1=etime(t2,t1);
e2=etime(t4,t3);
e3=etime(t4,t0);
fprintf('%s','Born-Kerbosch Algorithm time in minutes');
disp(e1/60);
fprintf('%s','Quadratic Solver time in minutes');
disp(e2/60);
fprintf('%s','total time in minutes');
disp(e3/60);
fprintf('%s','Independence number =');
disp(finalfval);
fprintf('%s','A maximum independent set =');
for i=1:novertices
    if finalx(i)>=0.9
```

```
        fprintf('%d,',i);
    end
end
fprintf('\n%s\n','************************')
end

Objective function code for 208 vertices graph.
function [f,gf,hf] = throughput(x)
f=-ones(1,208)*x.^2;
if nargout > 1 % gradient required
    gf=-2*x;
end
if nargout > 2 % Hessian required
    hf =-2*eye(208);
end
end
```

The code is three functions: (1) throughput function which is the objective function with its gradient, (2) the maximalCliques function which is not shown here downloaded from matlab code sharing community and (3) conversionA which prepares the the optimization problem parameters to be in the format for fmincon function and solves the problem using 100 uniformly distributed random seeds. By using matlab Parallel Add-on, a maximum independent set was yielded in 0.1451 minutes where the main time is due to fmincon of 0.1326 minutes and 0.0124 minutes to list maximal cliques.

Another important problem is finding an independent set of a certain size. This can be achieved by adding the constraint

$$\theta_1 + \theta_2 + ... + \theta_N = M \tag{6}$$

where M is the required independent set size.

7 Results Obtained form DIMACS Benchmarks

We see in table 1 and table 2 some of DIMACs maximum cliques benchmarks [14]. KBA stands for Bron-Kerbosch algorithm and SQP stands for sequential quadratic matlab algorithm. To find the maximum clique in these graphs, we found the maximum independent set in the complement graph. As can be seen our algorithm is efficient especially for sparse graphs that have short time to list their maximal cliques. Sequential quadratic programming is more accurate than interior-point and needs less runs. Since fmincon sometimes halts while trying to find a global maximum, we used in some graphs the equality constraint to find an independent set of that size which happens to be a maximum independent set for the graphs listed such as Keller4. In the second table for Keller4 graph the parallel processing version of fmincon halts responding in spite of finding the independence number in the first runs; for this we used a non-parallel fmincon

and the result shown in the table is for one such run. Fig. 5 shows the average running time when we use different powers of our polynomial formulation for 5 runs, each with 100 seeds for C125.9 graph [14], that has clique number equals to 34. It is clear increasing the polynomial power hasn't reduced execution time and as can be seen from Fig. 6 the accuracy is even less with more runs required to coverage to a global optima. We used a 500 seeds for a polynomial of degree 4 and we found independent sets of sizes 34 and 33, and with 1000 seeds for a polynomial of degree 10 we found an independent set of size 31. Now we come to some of the graphs where our algorithm performs better than cliquer [15]. One such graph is C250.9.clq [14] that has a clique number of 44. The maximal cliques listing time, quadratic programming algorithm time and total time were 0.1309, 125.6358 minutes (2.09393 hours), 125.7669 minutes (2.096115 hours) respectively by our algorithm to get a clique number of 44 while the maximum clique obtained by cliquer [15] is 39 after more than 7 hours of running. Indeed cliquer halted at 184/250 (max 39) 26202.11 (3330.10 s /round) and stopped responding. Another graph is P Gamma U34 On Nonisotropic Points [11], our quadratic programming algorithm yielded the independence number (clique number in the complement graph) 44 in 3.9597 minutes with 0.0182 minutes maximal cliques listing time, 3.9415 minutes for the quadratic solver fmincon; while cliquer took more than 7 hours without converging to the right independence number. Here is the last output of cliquer 146/208 (max 38) 25465.65 s (8904.73 s/round).

Table 1: DIMACS results using interior-point algorithm for fmincon.

Graph	***KBA time***	***No. of iterations***	***Interior-Point time***	ω
johnson8-2-4	0.0052	1	0.0075	4
MANN-a9	0.0013	5	0.0084	16
hamming6-2	0.0026	1	0.0072	32
hamming6-4	0.4790	10	23.2787	4
johson8-4-4	0.0937	20	0.1043	14
johnson16-2-4	0.0123	10	0.0827	8
C125.9	0.0097	100	0.6695	34
keller4 [a]	2.2866	10	12.8564	11

[a]with equality constraint, sum of θs=11

8 Conclusion

We proposed a quadratic optimization formulation to find maximal independent sets of any graph. We extended that to a polynomial optimization formulation. We need to list first the maximal cliques of the graph and then we solve a non-linear optimization problem. Our formulation is efficient when tested on some of the DIMAC maximum clique benchmarks and proved to be more efficient than a

Table 2: DIMACS results using sequential quadratic programming algorithm for fmincon.

Graph	***KBA time***	***No. of iterations***	***SQP time***	ω
johnson8-2-4	0.0035	1	0.0095	4
MANN-a9	0.0020	1	0.0062	16
hamming6-2	0.0030	1	0.0086	32
hamming6-4 [a]	0.4933	1	0.8052	4
johson8-4-4	0.0103	20	0.0206	14
johnson16-2-4	0.0130	10	0.0251	8
C125.9	0.0099	50	0.1152	34
keller4[b]	1.8339	1	2.8839	11

[a]with equality constraint, sum of θs=4

[b] run function stops responding, we used fmincon without parallel processing

Fig. 5: Average time of 5 runs, each with 100 seeds for different powers of the objective.

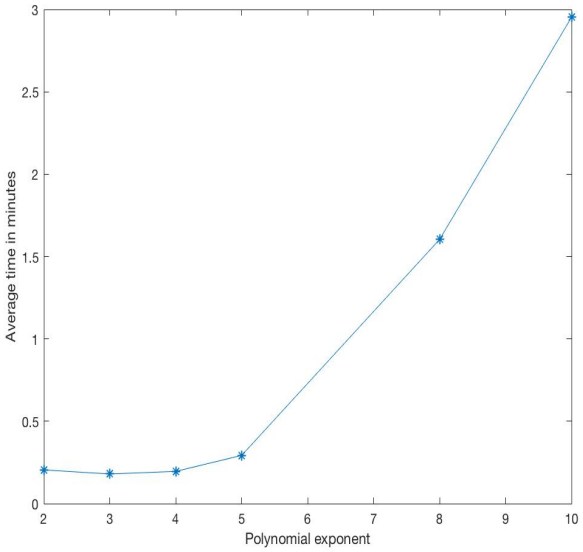
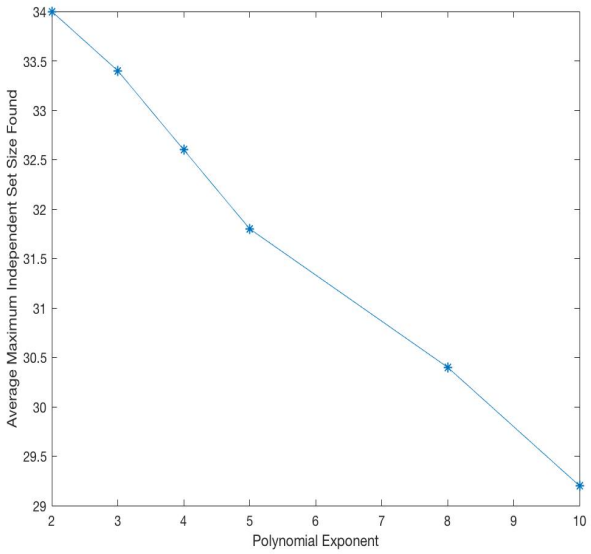


Fig. 6: Average Maximum Independent Set found of 5 runs, each with 100 seeds for different powers of the objective.



popular maximum clique algorithm such as cliquer for some graphs. However due to the time required to solve the quadratic or polynomial optimization problem, our algorithm works better when the listing time of maximal cliques of the graph is short, i.e. for sparse graphs. We can even reduce the time by coding an algorithm such as that in [13] to reduce maximal cliques listing time. This is to be tested in future for large sparse graph. The model can be extended easily to find an independent set of a certain size and to find the maximal weighted independent sets in weighted graphs.

Acknowledgment

Maher Heal thanks Dr. Una Benlic for advice on the maximal independent set and maximal clique problems while working on this research.

References

1. P. R. Östergård, "A fast algorithm for the maximum clique problem," *Discrete Applied Mathematics*, vol. 120, no. 1, pp. 197–207, 2002.
2. Z. Fang, C.-M. Li, and K. Xu, "An exact algorithm based on maxsat reasoning for the maximum weight clique problem," *Journal of Artificial Intelligence Research*, vol. 55, pp. 799–833, 2016.

3. W. A. Tavares, M. B. C. Neto, C. D. Rodrigues, and P. Michelon, "Um algoritmo de branch and bound para o problema da clique máxima ponderada," *Proceedings of XLVII SBPO*, vol. 1, 2015.
4. S. Shimizu, K. Yamaguchi, T. Saitoh, and S. Masuda, "Fast maximum weight clique extraction algorithm: Optimal tables for branch-and-bound," *Discrete Applied Mathematics*, vol. 223, pp. 120–134, 2017.
5. N. Z. Shor, "Dual quadratic estimates in polynomial and boolean programming," *Annals of Operations Research*, vol. 25, no. 1, pp. 163–168, 1990.
6. T. S. Motzkin and E. G. Straus, "Maxima for graphs and a new proof of a theorem of turán," *Canad. J. Math*, vol. 17, no. 4, pp. 533–540, 1965.
7. J. Harant, "Some news about the independence number of a graph," *Discussiones Mathematicae Graph Theory*, vol. 20, no. 1, pp. 71–79, 2000.
8. J. Harant, A. Pruchnewski, and M. Voigt, "On dominating sets and independent sets of graphs," *Combinatorics, Probability and Computing*, vol. 11, pp. 1–10, 1993.
9. P. Pardalos, L. Gibbons, and D. Hearn, "A continuous based heuristic for the maximum clique problem," Univ. of Michigan, Ann Arbor, MI (United States), Tech. Rep., 1994.
10. K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," *Wireless networks*, vol. 11, no. 4, pp. 471–487, 2005.
11. https://hog.grinvin.org/.
12. https://uk.mathworks.com/matlabcentral/fileexchange/30413-bron-kerbosch-maximal-clique-finding-algorithm.
13. D. Eppstein, M. Löffler, and D. Strash, "Listing all maximal cliques in sparse graphs in near-optimal time," in *International Symposium on Algorithms and Computation*. Springer, 2010, pp. 403–414.
14. http://iridia.ulb.ac.be/~fmascia/maximum_clique/.
15. https://users.aalto.fi/~pat/cliquer.html.