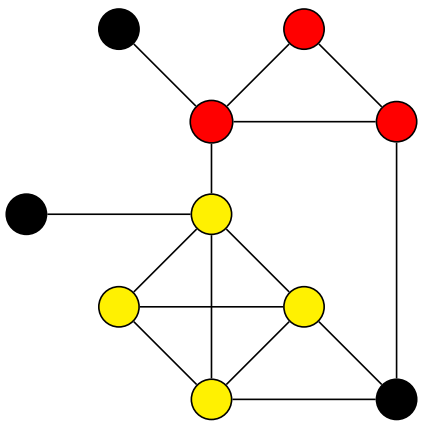
Why is maximum clique often easy in practice?

Jose L. Walteros (@TotalUnimodular)
University at Buffalo

Austin L. Buchanan (@AustinLBuchanan)
Oklahoma State University

What is a clique?

Luce and Perry (1949) introduced "clique" as ideal cluster in social network.
Also appears in bioinformatics, telecommunications, ...



Definition

A clique $C \subseteq V$ in graph $G = (V, E)$ is a subset of pairwise adjacent vertices.

Common task: find a *maximum* clique. Its size is denoted $\omega(G)$. Here, $\omega = 4$.

Issue: NP-hard (Cook, 1971), to get $n^{1-\varepsilon}$ approximation (Zuckerman, 2006).

Max clique can be hard

Some synthetic graphs from DIMACS2. Times from Prosser (2012).

graph	n	m	ω	time (s)
brock800_1	800	207,505	23	>14,400
brock800_2	800	208,166	24	>14,400
brock800_3	800	207,333	25	>14,400
hamming10-4	1,024	434,176	40	>14,400
johnson32-2-4	496	107,880	16	>14,400
keller5	776	225,990	27	>14,400
keller6	3,361	4,619,898	≥59	>14,400
MANN_a81	3,321	5,506,380	≥1,100	>14,400
p_hat700-3	700	183,010	≥62	>14,400
p_hat1000-3	1,000	371,746	≥68	>14,400
p_hat1500-2	1,500	568,960	≥65	>14,400
p_hat1500-3	1,500	847,244	≥94	>14,400

Max clique is often easy

Some real-life graphs from DIMACS10 & SNAP.

graph	n	m	ω	time (s)
coAuthorsDBLP	299,067	977,676	115	0.04
web-NotreDame	325,729	1,090,108	155	0.07
coPapersCiteseer	434,102	16,036,720	845	0.17
coPapersDBLP	540,486	15,245,729	337	0.20
web-BerkStan	685,230	6,649,470	201	0.25
eu-2005	862,664	16,138,468	387	0.50
in-2004	1,382,908	13,591,473	489	0.47
wiki-Talk	2,394,385	4,659,565	26	36.92
uk-2002	18,520,486	261,787,258	944	15.72

Why? What's different?

A closer look at synthetic instances

Graph degeneracy d is one measure of sparsity. A useful fact is $\omega \leq d+1$.

graph	n	m	ω	$d+1$	g	time (s)
brock800_1	800	207,505	23	488	465	>14,400
brock800_2	800	208,166	24	487	463	>14,400
brock800_3	800	207,333	25	484	459	>14,400
hamming10-4	1,024	434,176	40	849	809	>14,400
johnson32-2-4	496	107,880	16	436	420	>14,400
keller5	776	225,990	27	561	534	>14,400
keller6	3,361	4,619,898	≥59	2,691	≤2,632	>14,400
MANN_a81	3,321	5,506,380	≥1,100	3,281	≤2,181	>14,400
p_hat700-3	700	183,010	≥62	427	≤365	>14,400
p_hat1000-3	1,000	371,746	≥68	610	≤542	>14,400
p_hat1500-2	1,500	568,960	≥65	505	≤440	>14,400
p_hat1500-3	1,500	847,244	≥94	930	≤836	>14,400

Definition

The clique-core gap of a graph G is $g := (d+1) - \omega$.

A closer look at real-life instances

graph	n	m	ω	$d+1$	g	time (s)
coAuthorsDBLP	299,067	977,676	115	115	0	0.04
web-NotreDame	325,729	1,090,108	155	156	1	0.07
coPapersCiteseer	434,102	16,036,720	845	845	0	0.17
coPapersDBLP	540,486	15,245,729	337	337	0	0.20
web-BerkStan	685,230	6,649,470	201	202	1	0.25
eu-2005	862,664	16,138,468	387	389	2	0.50
in-2004	1,382,908	13,591,473	489	489	0	0.47
wiki-Talk	2,394,385	4,659,565	26	132	106	36.92
uk-2002	18,520,486	261,787,258	944	944	0	15.72

Observation: real-life graphs often have tiny g and are very easy.

- Is this pattern more than a coincidence?
- Must hard instances have large g?
- Are small g instances always easy?

YES!

Theorem

When $g = O(1)$, maximum clique solvable in $O(dm) = O(m^{1.5})$ time.

Overview of the rest of the talk

1. Background
   - degeneracy and k-cores
   - the connection with vertex cover
   - kernelization and fpt for vertex cover
2. Our proposed algorithm
3. Computational experiments

Degeneracy is a graph invariant that measures sparsity.

(degeneracy) = (largest k such that k-core exists) = (width) = (linkage)

Definition (Degeneracy)

The degeneracy of a graph G is the max-min-degree over all subgraphs G'.

$$d(G) = \max_{G' \subseteq G} \left\{ \min_{v \in V(G')} \{\deg_{G'}(v)\} \right\}.$$

- $d(\text{tree}) = 1$
- $d(C_n) = 2$
- $d(\text{planar}) \leq 5$
- $d(K_n) = n - 1$

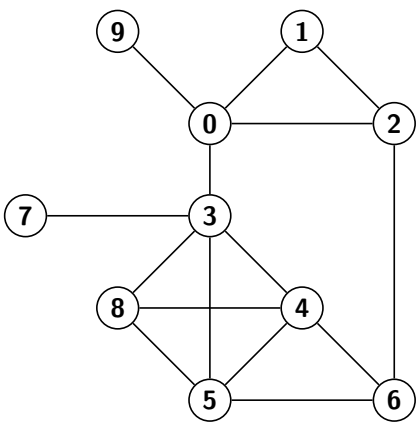
Remark

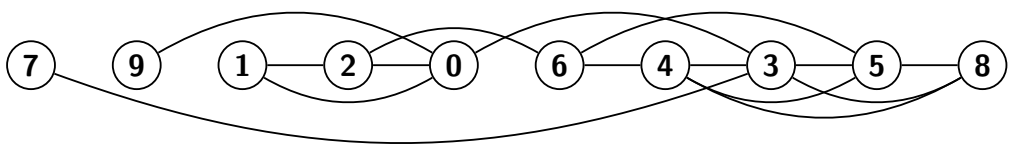
By the definition, $d(G) \geq \omega(G) - 1$. *In other words,* $\omega(G) \leq d(G) + 1$.

Computing degeneracy

Theorem (Matula and Beck, 1983)

Degeneracy can be computed in linear time by repeatedly removing a minimum degree vertex.



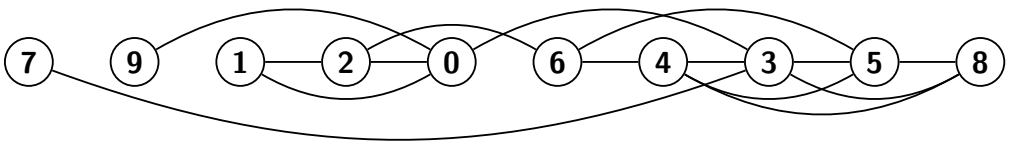


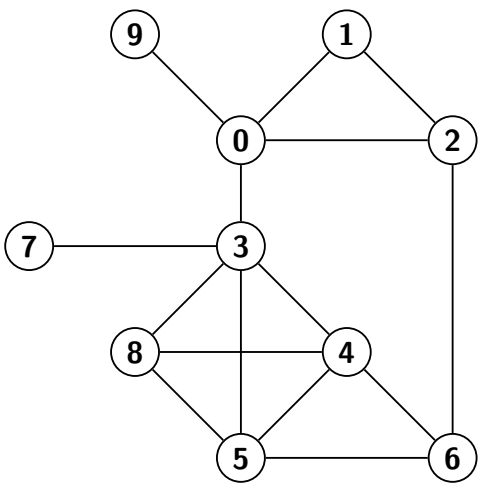
This gives a *minimum degree* (MD) *ordering* $(v_1, v_2, \ldots, v_n)$ of the vertices. The right-degree $\mathrm{rdeg}(v_i)$ of each vertex v_i is at most d. Importantly,

$$d = \max_i \{\mathrm{rdeg}(v_i)\}.$$

FPT algorithms for maximum clique

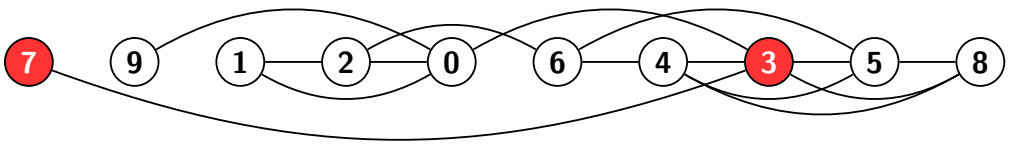
Each clique will belong to one of these (closed) right-neighborhoods!

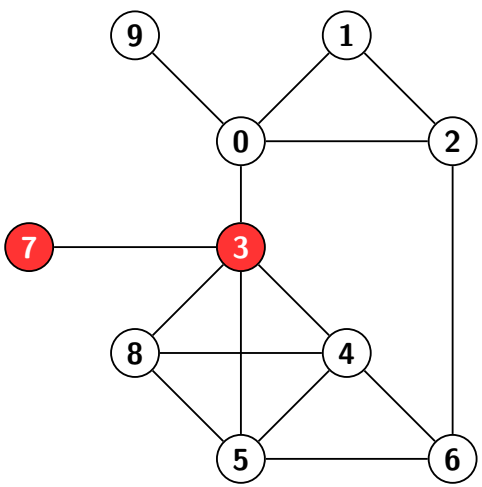




FPT algorithms for maximum clique

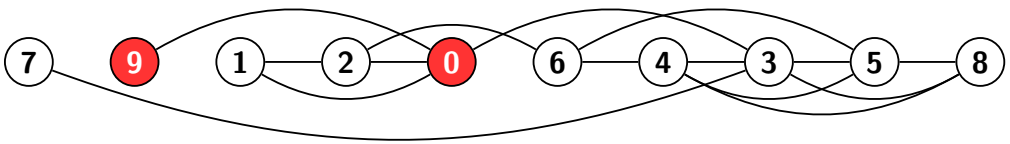
Each clique will belong to one of these (closed) right-neighborhoods!

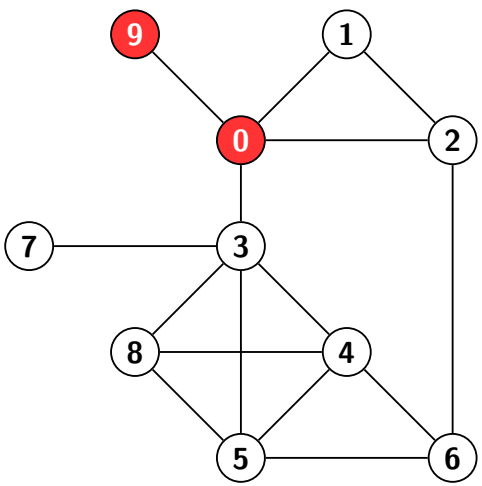




FPT algorithms for maximum clique

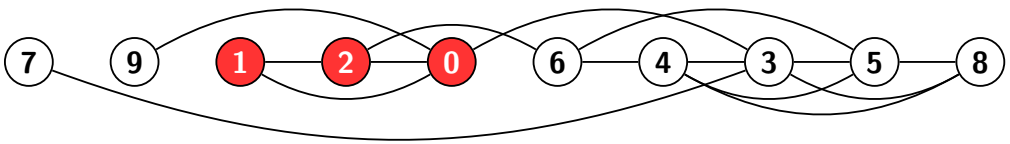
Each clique will belong to one of these (closed) right-neighborhoods!

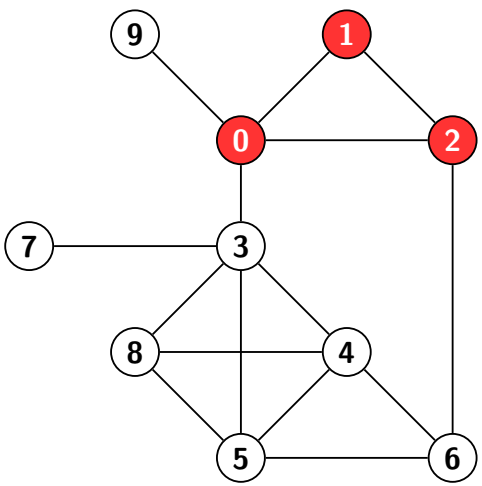




FPT algorithms for maximum clique

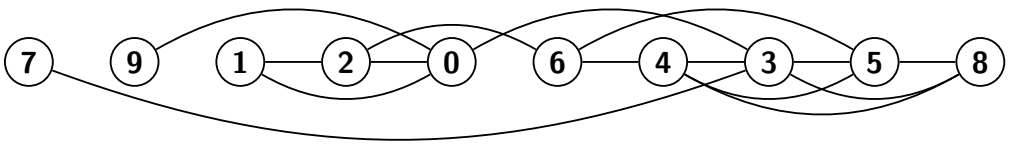
Each clique will belong to one of these (closed) right-neighborhoods!

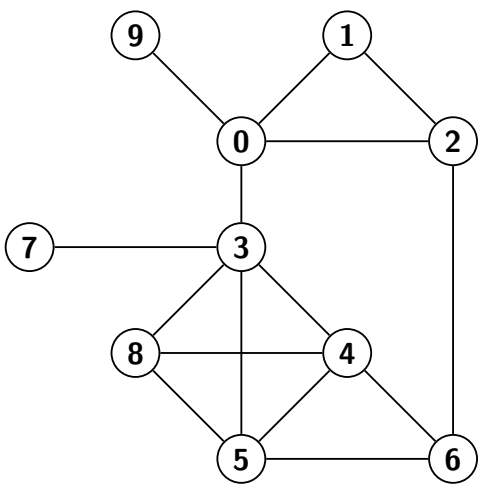


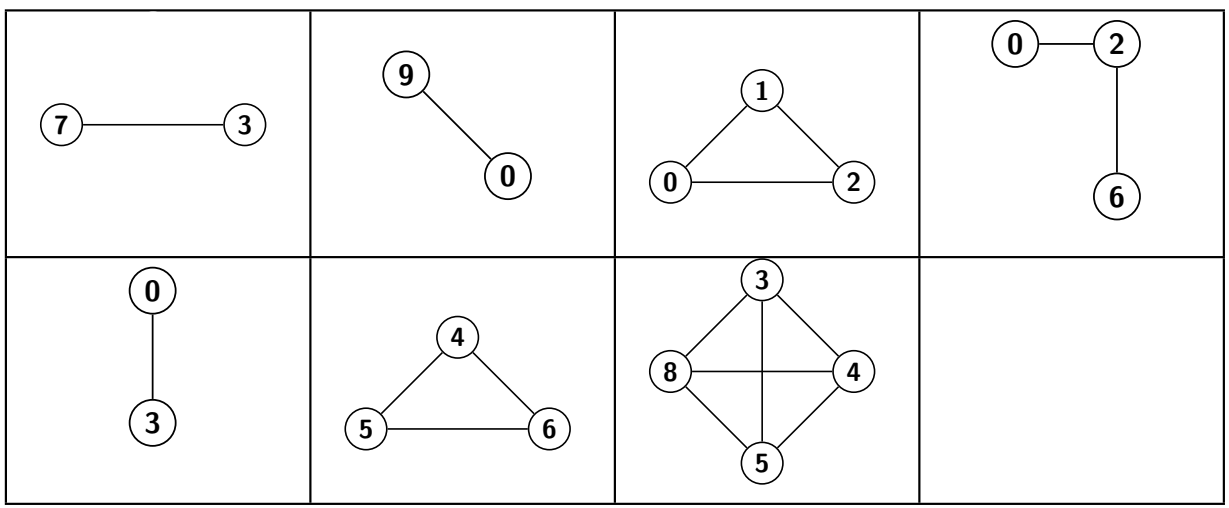


FPT algorithms for maximum clique

Each clique will belong to one of these (closed) right-neighborhoods!







Importantly, these right-neighborhoods are all "small"!

This insight used to:

- enumerate all maximal cliques
  - in time $O(dn3^{d/3})$ by Eppstein et al. (2010).
- find a maximum clique
  - in time $O(nm + n2^{d/4})$ by B et al. (2014).
  - in time $O((n-d)2^{d/4})$ by Manoussakis (2014).

These are fixed-parameter tractable (fpt) with respect to d, i.e., runtime of

$$f(d) \cdot \mathrm{poly}(n).$$

These might not be practical when $d \geq 100$.

Recall. . .

graph	n	m	ω	$d+1$	g	time (s)
coAuthorsDBLP	299,067	977,676	115	115	0	0.04
web-NotreDame	325,729	1,090,108	155	156	1	0.07
coPapersCiteseer	434,102	16,036,720	845	845	0	0.17
coPapersDBLP	540,486	15,245,729	337	337	0	0.20
web-BerkStan	685,230	6,649,470	201	202	1	0.25
eu-2005	862,664	16,138,468	387	389	2	0.50
in-2004	1,382,908	13,591,473	489	489	0	0.47
wiki-Talk	2,394,385	4,659,565	26	132	106	36.92
uk-2002	18,520,486	261,787,258	944	944	0	15.72

What would really be nice is fpt with respect to g!

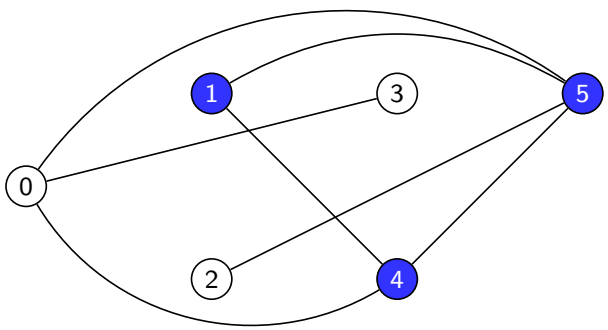
The Connection to Vertex Cover

Lemma

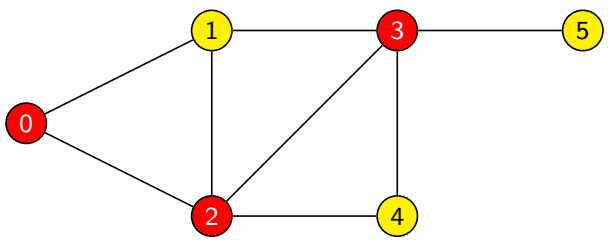
Letting α and τ denote the independence and vertex cover numbers,

$$\omega(G) = \alpha(\overline{G}) = n - \tau(\overline{G}).$$

G



$\overline{G}$



$$3 = 3 = 6 - 3.$$

Vertex Cover is FPT

Does $G = (V, E)$ have a vertex cover of size k?

Nemhauser-Trotter Kernel

1. Get optimal solution $x^* \in \left\{0, \frac{1}{2}, 1\right\}^n$ to

$$LP := \min_{x \geq 0} \left\{ \sum_{v \in V} x_v \;\middle|\; x_i + x_j \geq 1,\ \forall \{i,j\} \in E \right\},$$

2. If $LP > k$, then "no";
3. Else, work on the *kernel* $(G - F - Z, k - |F|)$, where

$$F = \{v \in V \mid x^*_v = 1\}$$
$$Z = \{v \in V \mid x^*_v = 0\}.$$

Importantly, kernel has at most $2k$ vertices! $\implies$ k-VC in time $O^*(2^{2k})$

With additional ideas, Chen et al. (2010) reduce time to $O(kn + 1.2738^k)$.

The Connection to Vertex Cover

Lemma

Letting α and τ denote the independence and vertex cover numbers,

$$\omega(G) = \alpha(\overline{G}) = n - \tau(\overline{G}).$$

Is the "FPT-ness" of Vertex Cover exploitable?

Some calculations

Does graph Wiki-Talk have a clique of size $d+1$? (Is $g = 0$?)

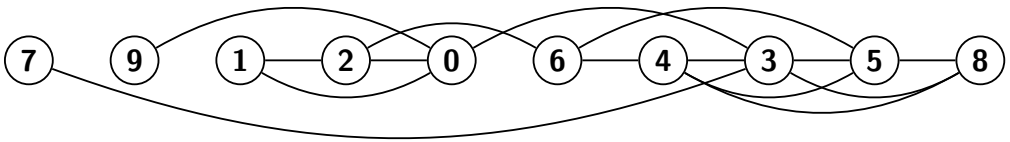
- $n = 2,394,385$
- $d + 1 = 132$

Equivalently, does $\overline{G}$ have a vertex cover of size k?

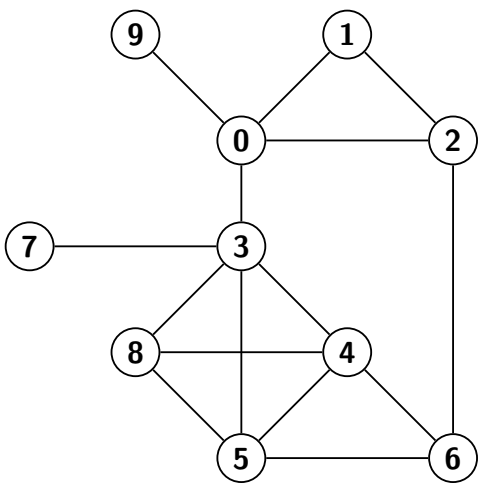
- $\overline{n} = 2,394,385$
- $\overline{m} \approx 3$ trillion
- $k = 2,394,253$

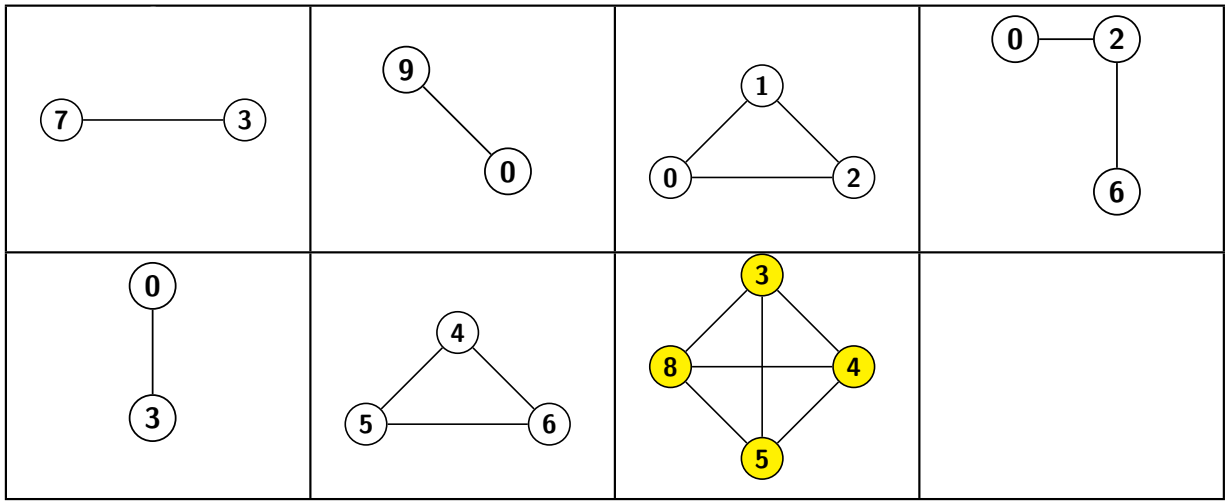
Back to the MD Ordering

Let V_i' be the closed right-neighborhood of v_i.



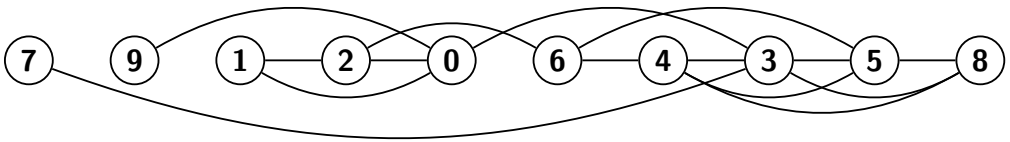
Does any subgraph $G[V_i']$ have a clique of size $d+1$? (Is $g = 0$?)



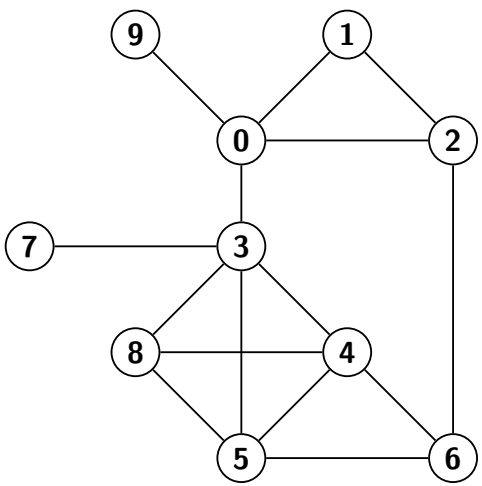


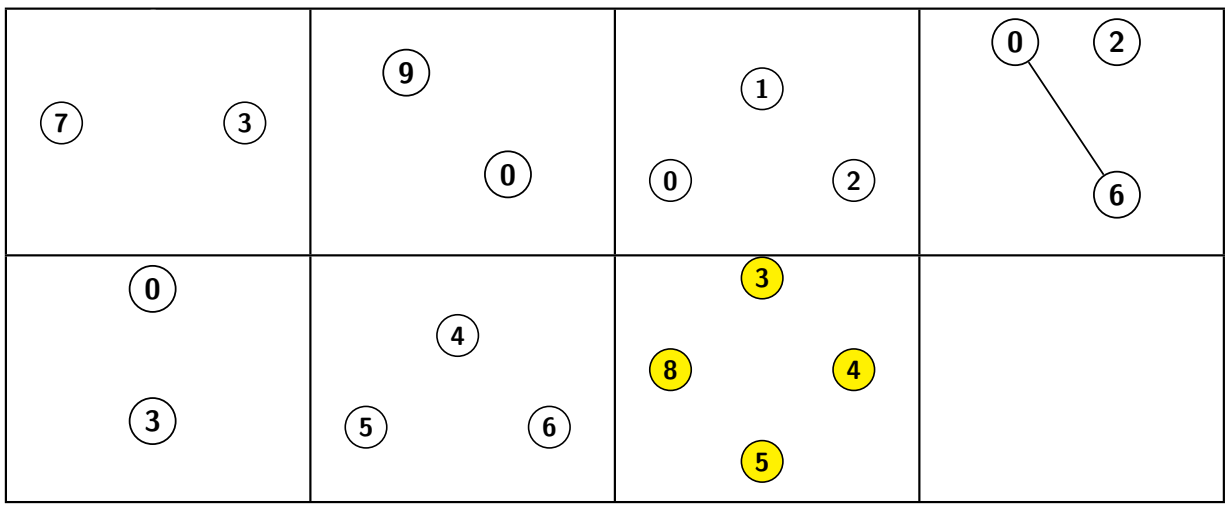
Back to the MD Ordering

Let V_i' be the closed right-neighborhood of v_i.



Equivalently, does any $\overline{G}[V_i']$ have a vertex cover of size $|V_i'| - (d+1)$?





Our Proposed Algorithm

Does G have a clique of size $(d+1)-p$? (Is $g \leq p$?)

main(G, p)

1. compute an MD ordering $(v_1, v_2, \ldots, v_n)$ and degeneracy d of G;
2. let $D = \{v_i \in V \mid i \leq n-d,\ \mathrm{rdeg}(v_i) \geq d-p\}$;
3. **for** $v_i \in D$ **do**
   - 3.1 construct $\overline{G}[V_i]$, where V_i is the open right-neighborhood of v_i;
   - 3.2 **if** $\overline{G}[V_i]$ has a vertex cover of size $q_i := |V_i| + p - d$, **return** "yes";
4. construct $\overline{G}[V_f]$, where $V_f = \{v_f, \ldots, v_n\}$ and $f := n - d + 1$;
5. if $\overline{G}[V_f]$ has a vertex cover of size $q_f := p - 1$, return "yes";
6. **return** "no".

Important observations: $|V_i| \leq d$ and $q_i \leq p$.

Main Complexity Results

Theorem

Algorithm **main** *determines whether graph G of degeneracy d has a clique of size $(d+1)-p$ in time $O((n-d)(1.28^p + d^2))$ and space $O(m + \mathrm{poly}(d))$.*

Theorem

When p is a constant, algorithm **main** *runs in time $O(dm) = O(m^{1.5})$.*

Remark: A linear-time special case

If $p = 0$ and the d-core of G is d-regular, then **main** runs in linear time

Corollary

Let $g := (d+1) - \omega$ be the clique-core gap. We can compute ω

1. *in time $1.28^g\,\mathrm{poly}(n)$;*
2. *in time $O(dm) = O(m^{1.5})$ when g is a constant;*
3. *in polynomial time when $g = O(\log n)$.*

Computational Experiments

- 60 instances, 50 from Verma et al. (2015), 10 from Rossi et al. (2015)
- DIMACS-10 and SNAP repositories
- Nine categories: (1) citation networks, (2) synthetic graphs, (3) peer-to-peer Internet networks, (4) online social networks, (5) web, (6) sparse matrices, (7) communication networks, (8) road networks, and (9) product networks

Computational Experiments

Type	Source	Name	n	m	Δ	d	ω	g
Citation	SNAP	Cit-HepTh	27,770	352,285	2,468	37	23	15
	SNAP	Cit-HepPh	34,546	420,877	846	30	19	12
	DIMACS_10	coAuthorsCiteseer	227,320	814,134	1,372	86	87	0
	DIMACS_10	citationCiteseer	268,495	1,156,647	1,318	15	13	3
	DIMACS_10	coAuthorsDBLP	299,067	977,676	336	114	115	0
	DIMACS_10	coPapersCiteseer	434,102	16,036,720	1,188	844	845	0
	DIMACS_10	coPapersDBLP	540,486	15,245,729	3,299	336	337	0
	DIMACS_10	er-fact1.5-scale20	1,048,576	10,904,496	45	14	3	12
	SNAP	cit-Patents	3,774,768	16,518,947	793	64	11	54
Synthetic	DIMACS_10	delaunay_n16	65,536	196,575	17	4	4	1
	DIMACS_10	kron_g500-simple-logn16	65,536	2,456,071	17,997	432	136	297
	DIMACS_10	preferentialAttachment	100,000	499,985	983	5	6	0
	DIMACS_10	G_n_pin_pout	100,000	501,198	25	7	4	4
	DIMACS_10	smallworld	100,000	499,998	17	7	6	2
	DIMACS_10	delaunay_n17	131,072	393,176	17	4	4	1
	DIMACS_10	delaunay_n18	262,144	786,396	21	4	4	1
	DIMACS_10	rgg_n_2_21_s0	2,097,152	14,487,995	37	18	19	0
	DIMACS_10	rgg_n_2_22_s0	4,194,304	30,359,198	36	19	20	0
	DIMACS_10	rgg_n_2_23_s0	8,388,608	63,501,393	40	20	21	0
	DIMACS_10	rgg_n_2_24_s0	16,777,216	89,345,197	40	20	21	0
Internet topology and peer-to-peer	SNAP	p2p-Gnutella04	10,876	39,994	103	7	4	4
	SNAP	p2p-Gnutella25	22,687	54,705	66	5	4	2
	DIMACS_10	as-22july06	22,963	48,436	2,390	25	17	9
	SNAP	p2p-Gnutella24	26,518	65,369	355	5	4	2
	SNAP	p2p-Gnutella30	36,682	88,328	55	7	4	4
	SNAP	p2p-Gnutella31	62,586	147,892	95	6	4	3
	DIMACS_10	caidaRouterLevel	192,244	609,066	1,071	32	17	16
	SNAP	as-skitter	1,696,415	11,095,298	35,455	111	67	45

Computational Experiments

Type	Source	Name	n	m	Δ	d	ω	g
Social	SNAP	wiki-Vote	7,115	100,762	1,065	53	17	37
	SNAP	soc-Epinions1	75,879	405,740	3,044	67	23	45
	SNAP	soc-Slashdot0811	77,360	469,180	2,539	54	26	29
	SNAP	soc-Slashdot0922	82,168	504,230	2,552	55	27	29
	SNAP	soc-pokec	1,632,803	22,301,964	14,854	47	29	19
	SNAP	soc-LiveJournal1	4,847,571	42,851,237	20,333	372	321	52
Web	SNAP	web-Stanford	281,903	1,992,636	38,625	71	61	11
	DIMACS_10	cnr-2000	325,557	2,738,969	18,236	83	84	0
	SNAP	web-NotreDame	325,729	1,090,108	10,721	155	155	1
	SNAP	web-BerkStan	685,230	6,649,470	84,230	201	201	1
	DIMACS_10	eu-2005	862,664	16,138,468	68,963	388	387	2
	SNAP	web-Google	875,713	4,322,051	6,332	44	44	1
	DIMACS_10	in-2004	1,382,908	13,591,473	21,869	488	489	0
	SNAP	wiki-topcats	1,791,489	25,444,207	238,342	99	39	61
	DIMACS_10	uk-2002	18,520,486	261,787,258	194,955	943	944	0
Sparse matrices	DIMACS_10	wave	156,317	1,059,331	44	8	6	3
	DIMACS_10	audikw1	943,695	38,354,076	344	47	36	12
	DIMACS_10	ldoor	952,203	22,785,136	76	34	21	14
	DIMACS_10	ecology1	1,000,000	1,998,000	4	2	2	1
	DIMACS_10	333SP	3,712,815	11,108,633	28	4	4	1
	DIMACS_10	cage15	5,154,859	47,022,346	46	25	6	20
Road	DIMACS_10	luxembourg.osm	114,599	119,666	6	2	3	0
	DIMACS_10	belgium.osm	1,441,295	1,549,970	10	3	3	1

Computational Experiments

Type	Source	Name	n	m	Δ	d	ω	g
Communication	SNAP	email-Enron	36,692	183,831	1,383	43	20	24
	DIMACS_10	cond-mat-2005	40,421	175,691	278	29	30	0
	SNAP	email-EuAll	265,214	364,481	7,636	37	16	22
	SNAP	wiki-Talk	2,394,385	4,659,565	100,029	131	26	106
	SNAP	com-Orkut	3,072,441	117,185,083	33,313	253	51	203
Product co-purchasing	SNAP	Amazon0302	262,111	899,792	420	6	7	0
	SNAP	Amazon0312	400,727	2,349,869	2,747	10	11	0
	SNAP	Amazon0601	403,394	2,443,408	2,752	10	11	0
	SNAP	Amazon0505	410,236	2,439,437	2,760	10	11	0

Computational Experiments

14 out of 60 instances are solved in linear time

Property	Name	n	m	d	ω	g	Time
d-regular d-core	cond-mat-2005	40,421	175,691	29	30	0	0.00
	coAuthorsCiteseer	227,320	814,134	86	87	0	0.03
	coAuthorsDBLP	299,067	977,676	114	115	0	0.04
	coPapersCiteseer	434,102	16,036,720	844	845	0	0.17
	coPapersDBLP	540,486	15,245,729	336	337	0	0.20
	rgg_n_2_21_s0	2,097,152	14,487,995	18	19	0	0.55
	rgg_n_2_22_s0	4,194,304	30,359,198	19	20	0	1.33
	uk-2002	18,520,486	261,787,258	943	944	0	15.72
$l = d + 1$	preferentialAttachment	100,000	499,985	5	6	0	0.03
	cnr-2000	325,557	2,738,969	488	489	0	0.08
	Amazon0312	400,727	2,349,869	10	11	0	0.17
	in-2004	1,382,908	13,591,473	5	6	0	0.47
	rgg_n_2_23_s0	8,388,608	63,501,393	20	21	0	6.13
	rgg_n_2_24_s0	16,777,216	89,345,197	20	21	0	9.30

33 out of the other 46 are solved in under 1 second

Computational Experiments

All times obtained using same cluster:

- Linux x86_64, CentOS 7.1
- 12-core Intel Xeon E5-2640 v3 2.4GHz
- 128 GB RAM
- times exclude file I/O

Comparisons with single-thread implementations of:

(BWBP) Buchanan, Walteros, Butenko, Pardalos (Optimization Letters, 2014)

(VBB) Verma, Buchanan, Butenko (INFORMS Journal on Computing, 2015)

(RGG) Rossi, Gleich, Gebremedhin (SIAM Journal on Scientific Computing, 2015)

Results for parallel implementations are similar.

Computational Experiments

Instance	d	ω	g	**main**	BWBP	VBB	RGG
Cit-HepTh	30	19	12	0.18	0.34	**0.12**	0.19
Cit-HepPh	37	23	15	0.20	0.37	**0.12**	0.20
citationCiteseer	15	13	3	**0.11**	0.14	0.53	0.26
er-fact1.5-scale20	14	3	12	9.12	13.64	17.15	**6.87**
cit-Patents	64	11	54	**4.52**	5.29	12.79	5.89
delaunay_n16	4	4	1	**0.05**	0.08	0.22	0.09
kron_g500	432	136	297	2,663.15	815.16	335.94	**13.31**
G_n_pin_pout	7	4	4	0.27	0.50	0.42	**0.23**
smallworld	7	6	2	**0.10**	0.27	0.41	0.22
delaunay_n17	4	4	1	**0.11**	0.16	0.22	0.18
delaunay_n18	4	4	1	**0.23**	0.33	0.45	0.36
p2p-Gnutella04	7	4	4	**0.01**	0.03	**0.01**	0.02
p2p-Gnutella25	5	4	2	**0.01**	0.03	0.44	0.02
as-22july06	25	17	9	**0.01**	**0.01**	**0.01**	**0.01**
p2p-Gnutella24	5	4	2	**0.01**	0.04	0.69	0.02
p2p-Gnutella30	7	4	4	**0.02**	0.06	0.12	0.03
p2p-Gnutella31	6	4	3	**0.04**	0.10	0.07	0.05
caidaRouterLevel	32	17	16	**0.09**	0.12	0.23	0.15
as-skitter	111	67	45	**1.68**	2.57	6.21	2.74

Computational Experiments

Instance	d	ω	g	**main**	BWBP	VBB	RGG
wiki-Vote	53	17	37	0.27	0.21	0.10	**0.09**
soc-Epinions1	67	23	45	0.60	0.56	**0.26**	0.31
soc-Slashdot0811	54	26	29	0.29	0.33	**0.15**	0.20
soc-Slashdot0922	55	27	29	0.30	0.34	**0.16**	0.22
soc-pokec	47	29	19	13.50	12.93	19.45	**11.20**
soc-LiveJournal1	372	321	52	**4.95**	*	*	9.07
web-Stanford	71	61	11	**0.15**	0.25	2.96	0.53
web-NotreDame	155	155	1	**0.07**	0.09	0.12	0.23
web-BerkStan	201	201	1	**0.25**	0.29	6.96	1.33
eu-2005	388	387	2	**0.50**	0.72	26.29	3.06
web-Google	44	44	1	**0.33**	0.35	1.15	0.82
wiki-topcats	99	39	61	**12.58**	14.94	*	20.04
wave	8	6	3	**0.33**	1.39	0.63	0.49
audikw1	47	36	12	34.30	92.87	**14.10**	32.12
ldoor	34	21	14	19.04	36.58	**10.79**	14.36
ecology1	2	2	1	**0.55**	2.76	1.05	1.04
333SP	4	4	1	**1.49**	1.86	50.24	5.79
cage15	25	6	20	32.77	56.83	**23.38**	27.49

Computational Experiments

Instance	d	ω	g	**main**	BWBP	VBB	RGG
luxembourg.osm	2	3	0	0.02	0.02	0.18	**0.01**
belgium.osm	3	3	1	0.29	0.29	0.89	**0.28**
email-Enron	43	20	24	0.12	0.16	0.12	**0.09**
email-EuAll	37	16	22	**0.09**	0.11	0.12	0.12
wiki-Talk	131	26	106	36.92	**3.89**	5.82	5.25
com-Orkut	253	51	203	302.99	275.67	*	**241.00**
Amazon0302	6	7	0	**0.09**	**0.09**	0.32	0.17
Amazon0601	10	11	0	0.18	0.19	1.95	**0.13**
Amazon0505	10	11	0	0.18	0.19	2.19	**0.13**

Computational Experiments

We report shifted geometric means (unscaled and scaled) of single-threaded implementations. The shifting parameter was set to 1 second.

		main		BWBP		VBB		RGG	
Instance Class	Graphs	Unscaled	Scaled	Unscaled	Scaled	Unscaled	Scaled	Unscaled	Scaled
$g \in [0,1)$	18	**0.73**	**1.00**	1.55	2.11	3.06	4.17	0.96	1.31
$g \in [1,10)$	20	**0.21**	**1.00**	0.36	1.75	1.12	5.37	0.47	2.25
$g \in [10,100)$	19	**2.55**	**1.00**	5.09	1.99	3.36	2.49	2.65	1.04
$g \in [100,1000)$	3	312.15	11.62	102.36	3.81	201.30	7.49	**26.87**	**1.00**
Real-life	49	**1.30**	**1.00**	2.06	1.58	4.29	3.30	1.43	1.10
All	60	1.50	1.02	2.28	1.55	3.80	2.58	**1.48**	**1.00**

Conclusion

Summary

- Our aim: Why is max clique often easy?
- Our explanation: Small clique-core gap g is common and *exploitable*
- Uses results from FPT literature
- Competitive with previous algorithms
  - And we have worst-case guarantees!

Open questions

1. Can we test $g = 0$ in linear time?
   - Our approach takes time $\Omega(m^{1.5})$ on bipartite graphs $K_{d,2d}$.
2. Is there a practical parameterization for coloring?

References

- R. D. Luce and A. D. Perry. A method of matrix analysis of group structure. *Psychometrika*, 1949.
- S. A. Cook. The complexity of theorem-proving procedures. *Proceedings of the third annual ACM STOC*, 1971.
- D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Proceedings of the thirty-eighth annual ACM STOC*, 2006.
- P. Prosser. Exact algorithms for maximum clique: A computational study. *Algorithms*, 2012.
- D. W. Matula and L. L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM*, 1983.
- D. Eppstein, M. Löffler, and D. Strash. Listing all maximal cliques in sparse graphs in near-optimal time. *Algorithms and Computation*, 2010.
- A. Buchanan, J. L. Walteros, S. Butenko, P. M. Pardalos. Solving maximum clique in sparse graphs: an $O(nm + n2^{d/4})$ algorithm for d-degenerate graphs. *Optimization Letters*, 2014.
- G. Manoussakis. The clique problem on k-inductive graphs. arXiv preprint, 2014.
- G. L. Nemhauser and L. E. Trotter Jr. Vertex packings: structural properties and algorithms. *Mathematical Programming*, 1975.
- J. Chen, H. Fernau, I. A. Kanj, G. Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 2010.
- A. Verma, A. Buchanan, S. Butenko. Solving the maximum clique and vertex coloring problems on very large sparse networks. *INFORMS Journal on Computing*, 2015.
- R. A. Rossi, D. F. Gleich, A. H. Gebremedhin. Parallel maximum clique algorithms with applications to network analysis. *SIAM Journal on Scientific Computing*, 2015.