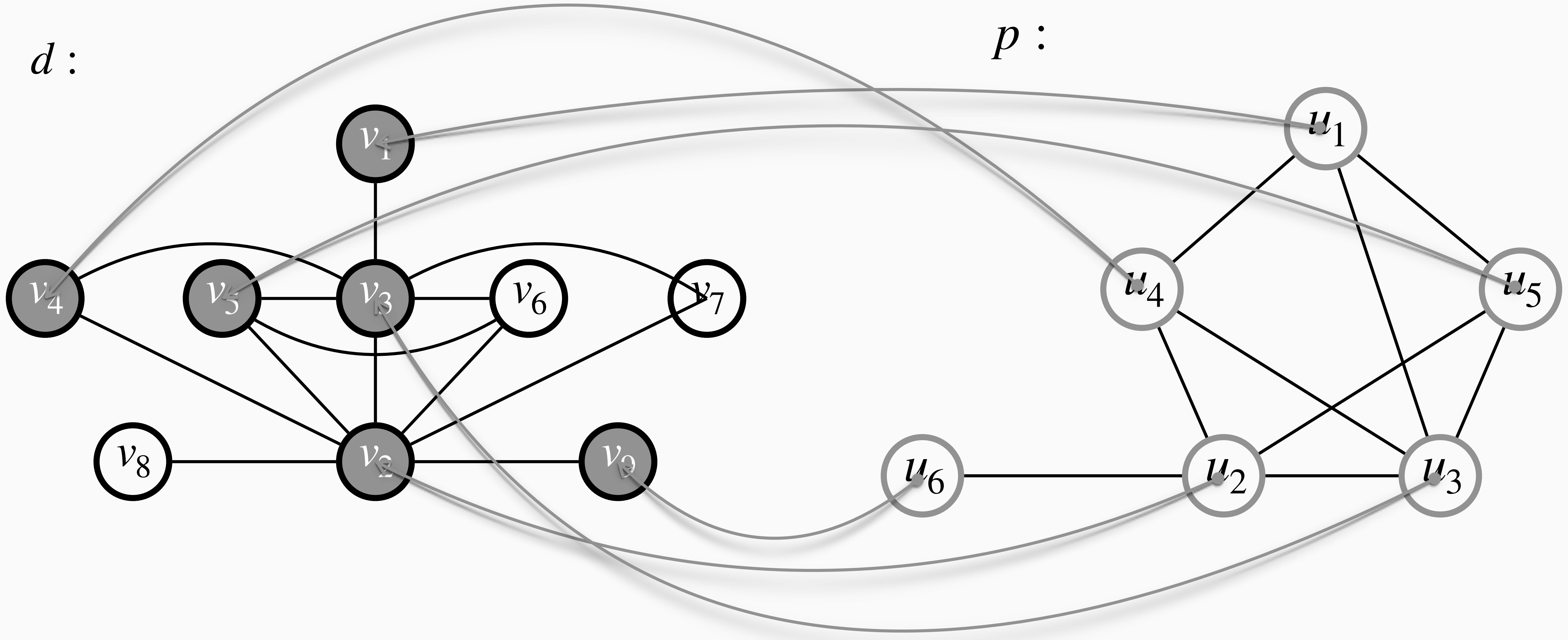
Subgraph Matching: on Compression and Computation

Miao Qiao′, Hao Zhang*, Hong Cheng*

The University of Auckland′
The Chinese University of Hong Kong*

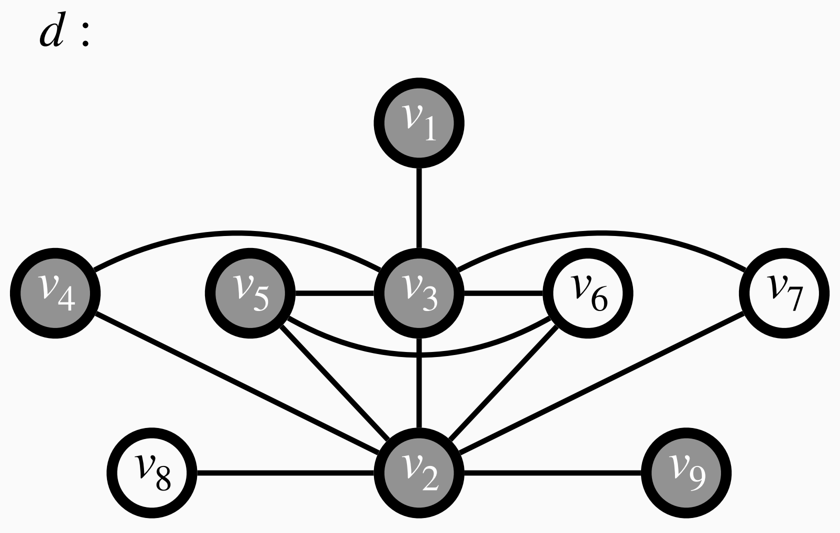
Subgraph Matching. Given a target graph *d* and a pattern graph *p*, report the set *I* of all the subgraphs in the target graph *d* that are isomorphic to the pattern graph *p*.

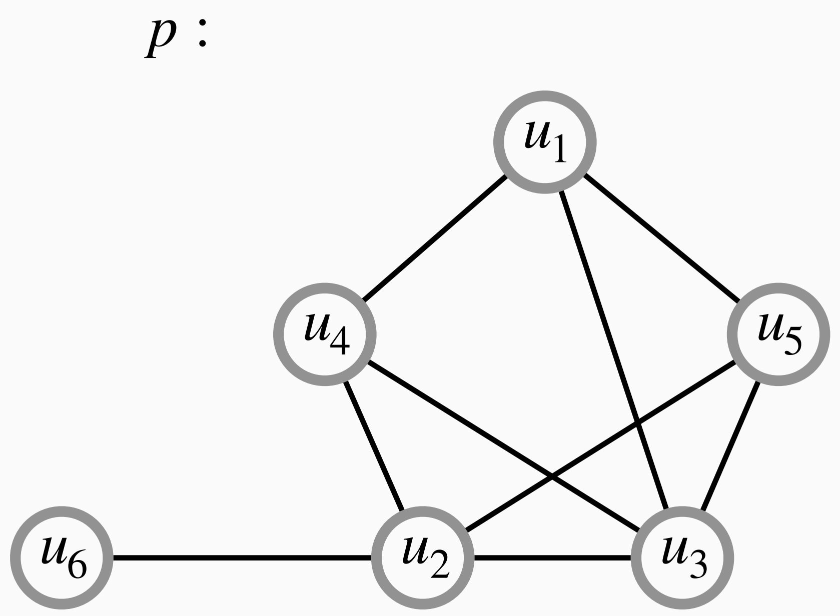


$$f: u_1 \mapsto v_1, u_2 \mapsto v_2, \cdots, u_5 \mapsto v_5, u_6 \mapsto v_9$$

Motivation

Output Crisis





dataset	size(d) in MB	P's Match Result
ego-Gplus(GP)	390	270.15TB
web-BerkStan(WB)	211	14.45TB
as-Skitter(AS)	355	8.46TB
soc-LiveJournal(LJ)	1373	9.14TB
uk-2002(UK)	9539	246.59TB

Current Method

- No output : repetitive computation
- Output the resulting set : expensive I/O cost

Motivation

Current Method

- No output : repetitive computation
- Output the resulting set : expensive I/O cost

Output solution: Is there an ideal compression on the instance set I ?

Lossless:

The compression should be lossless

d-optimal:

The decompression should be optimal, that is, the resulting set I of a subgraph matching restored in $O(\frac{|I|}{B})$ I/Os.

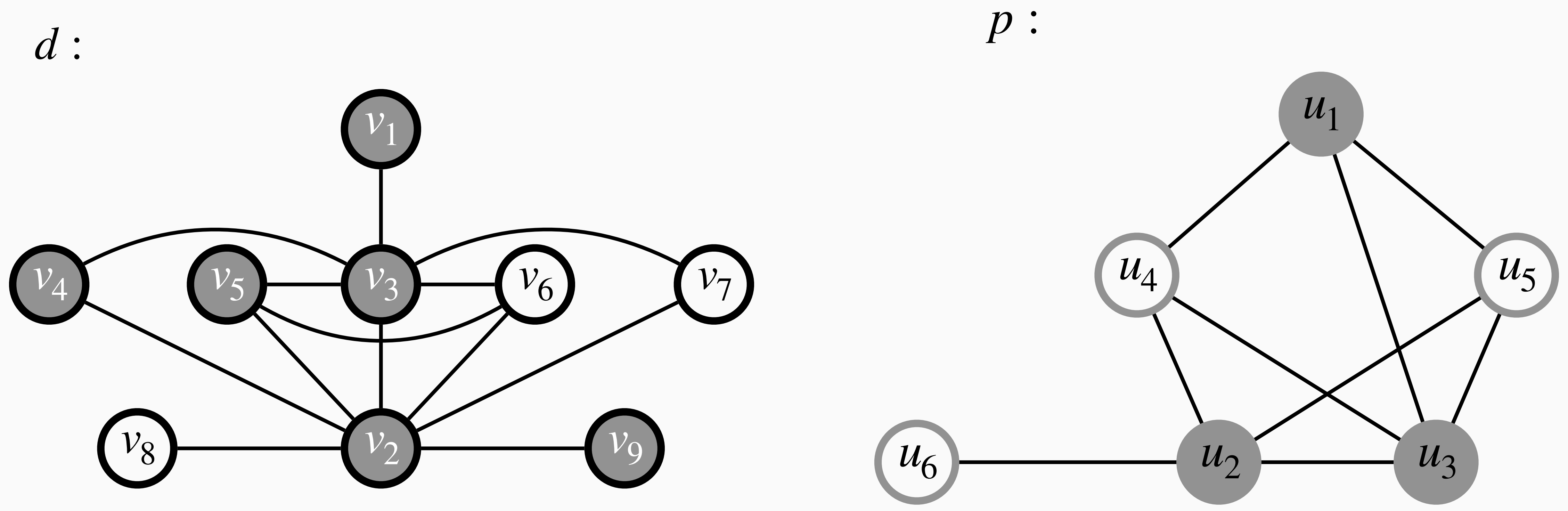
Motivation

Possible Bijection between p and d given a image for u1, u2, u3(Uncompressed)

d\p	u_1	u_2	u_3	u_4	u_5	u_6
v_1	■					
v_2		■				
v_3			■			
v_4				■		■
v_5				■	■	■
v_6				■	■	■
v_7					■	■
v_8						■
v_9						■

$|u_4| \times |u_5| \times |u_6|$

54 Possible Combination ! = 54*6*4 bytes



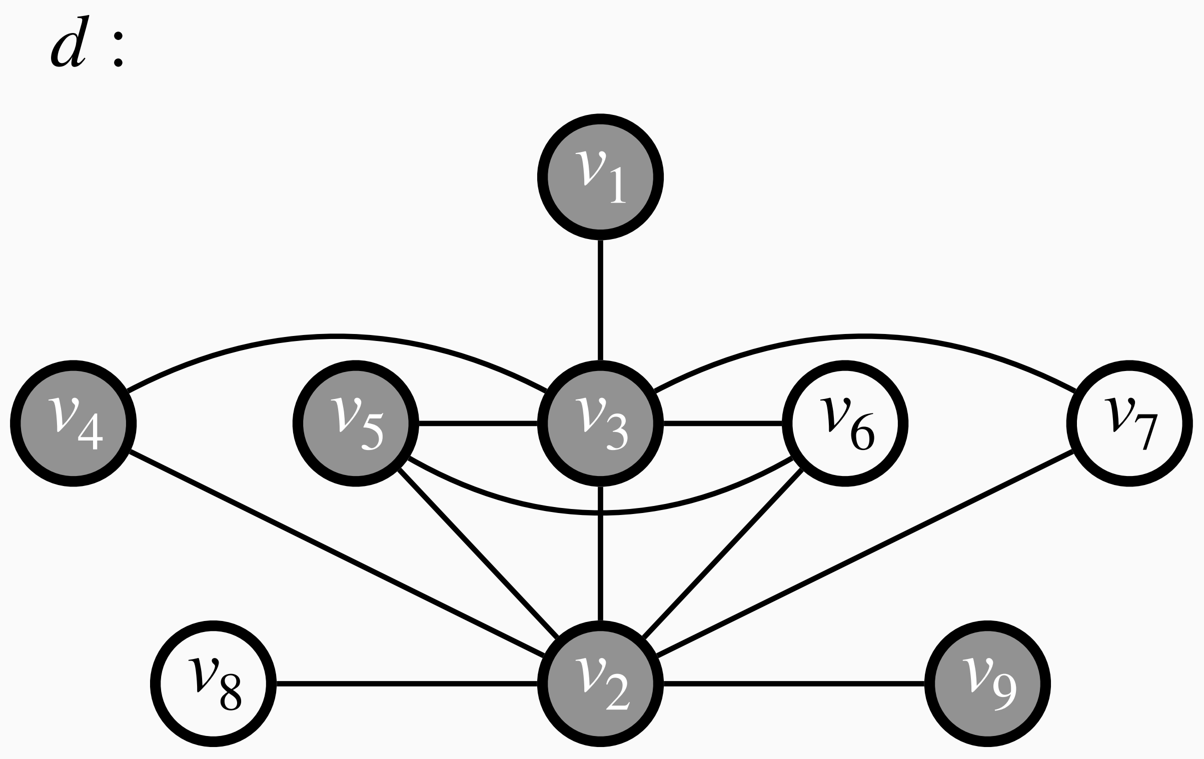
Possible Bijection between p and d given bijection for u1, u2, u3(Compressed)

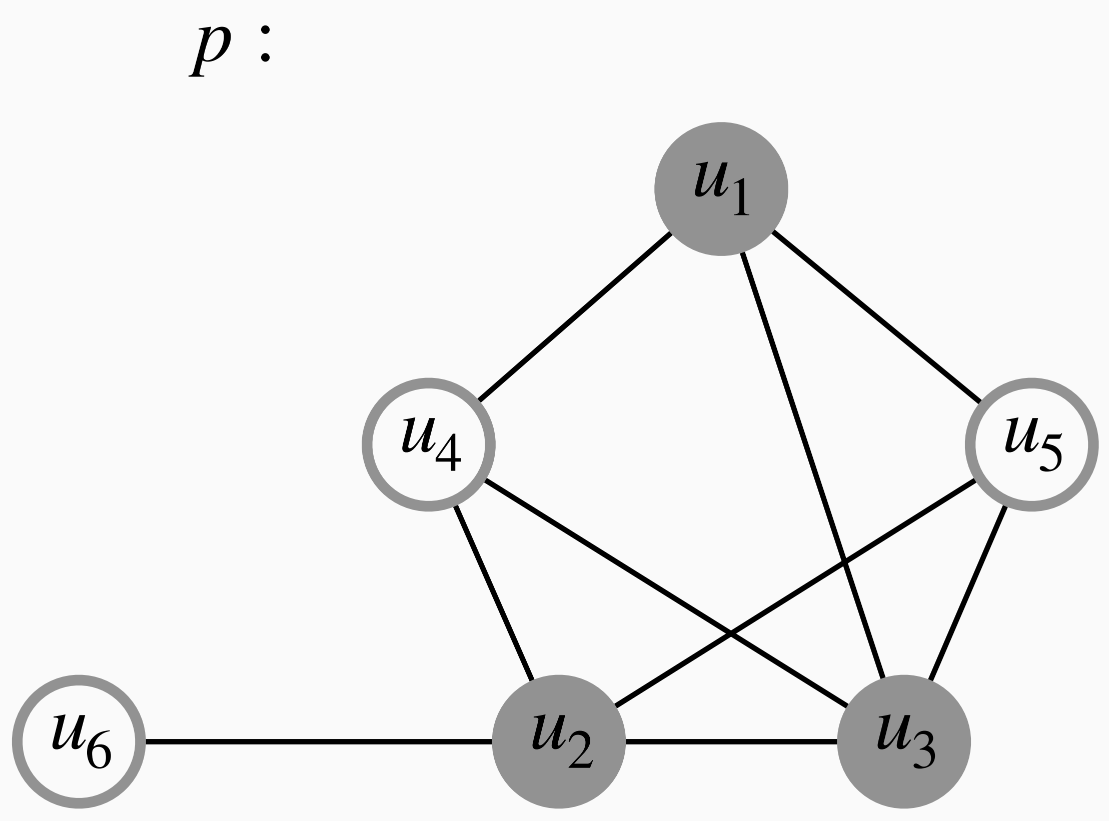
V(p)	u_1	u_2	u_3	u_4	u_5	u_6
Images	v_1	v_2	v_3	v_4, v_5, v_6	v_5, v_6, v_7	$v_4, v_5, \ldots, v_9$

Only (3+(3+3+6))*4 bytes

Vertex-Cover Based Compression(VCBC)

- Identify a vertex-cover Vc of p
- Group the matched subgraphs by their images on the vertex cover nodes
- Compress : list the image sets of each node in V(p) for each group





Theorem(Lossless)

The vertex-cover based compression is lossless, that is, for a given V_c , $\bar{I} = I$.

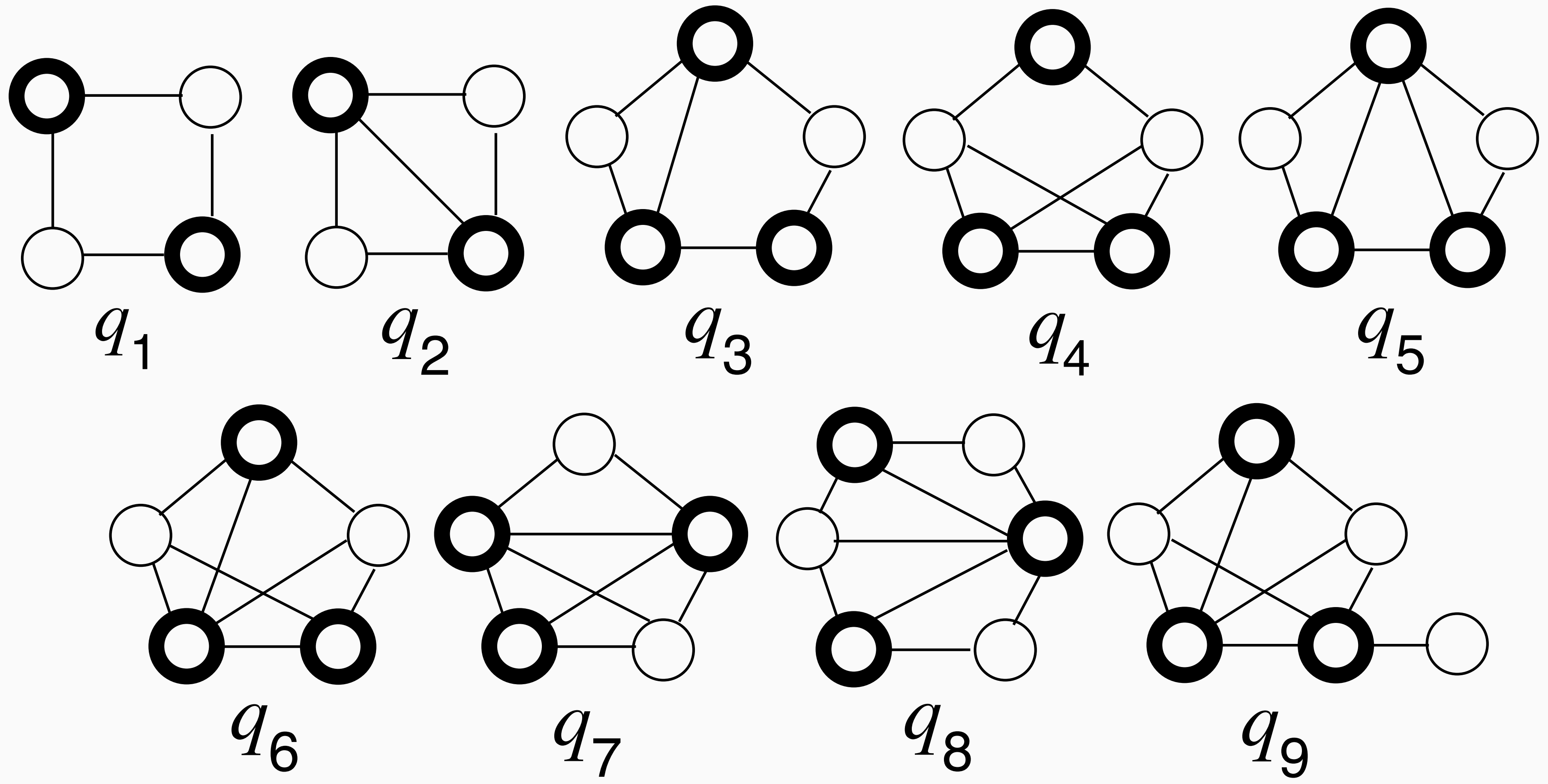
Theorem(d-optimal)

The vertex-cover based compression is decompression optimal, that is, the resulting set I of a subgraph matching can be restored in $O(\frac{|I|}{B})$ I/Os.

Experiment TestCase

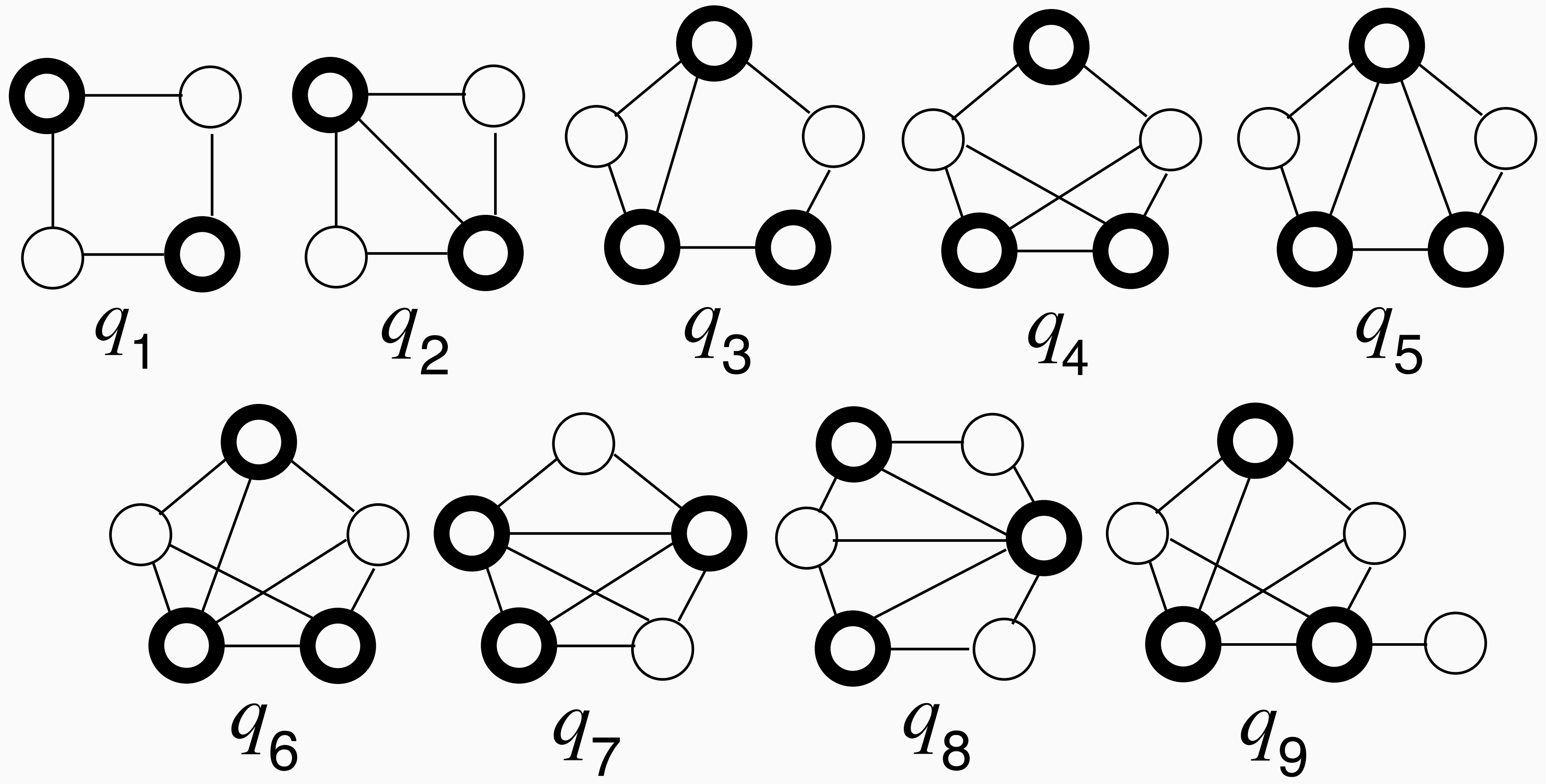
dataset	|V(d)| * 10^6	|E(d)| * 10^6	avg-deg	Degen-Eracy	size(d) in MB
ego-Gplus(GP)	0.1	12.2	244	1504	390
web-BerkStan(WB)	0.7	6.6	19	402	211
as-Skitter(AS)	1.7	11.1	13	222	355
soc-LiveJournal(LJ)	4.8	42.9	18	746	1373
uk-2002(UK)	18.5	298.1	32	1886	9539

Target Graph



Pattern Graph

Experiment Compression

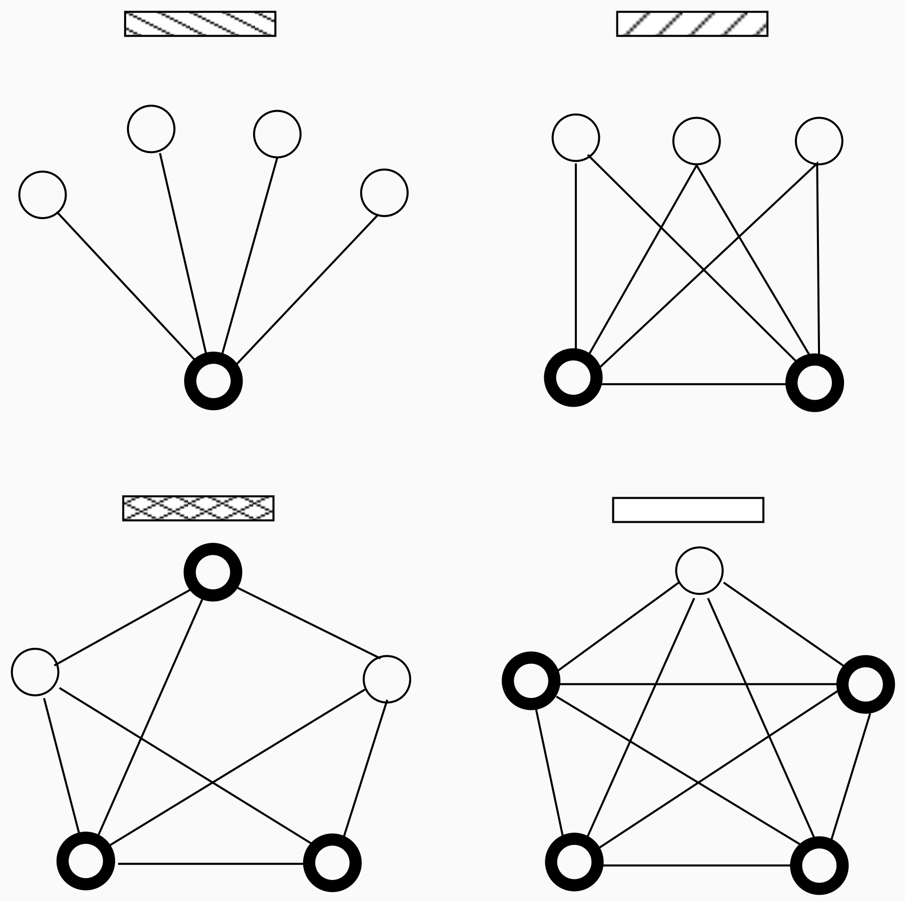


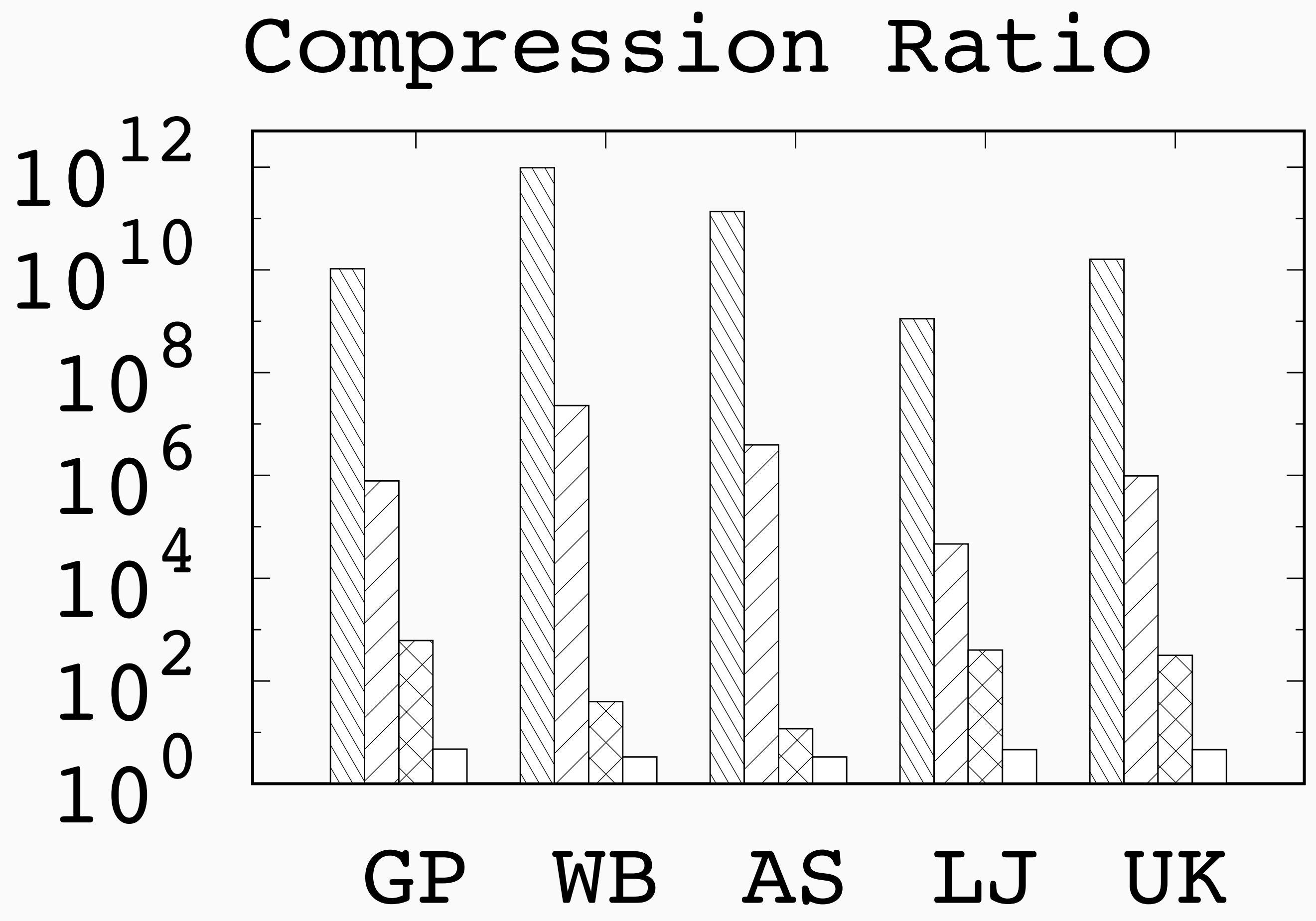
Pattern Graph

Compression Ratio

d\p	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
GP	333	1435	1263	409	1016	601	862	636433	23871
WB	17	2031	93	27	107	39	127	176833	93842
AS	23	790	80	9	76	12	66	39979	12724
LJ	19	342	581	201	362	400	440	147317	45336
UK	40	787	350	156	348	315	483	238077	130367

Compression Ratio - Freedom





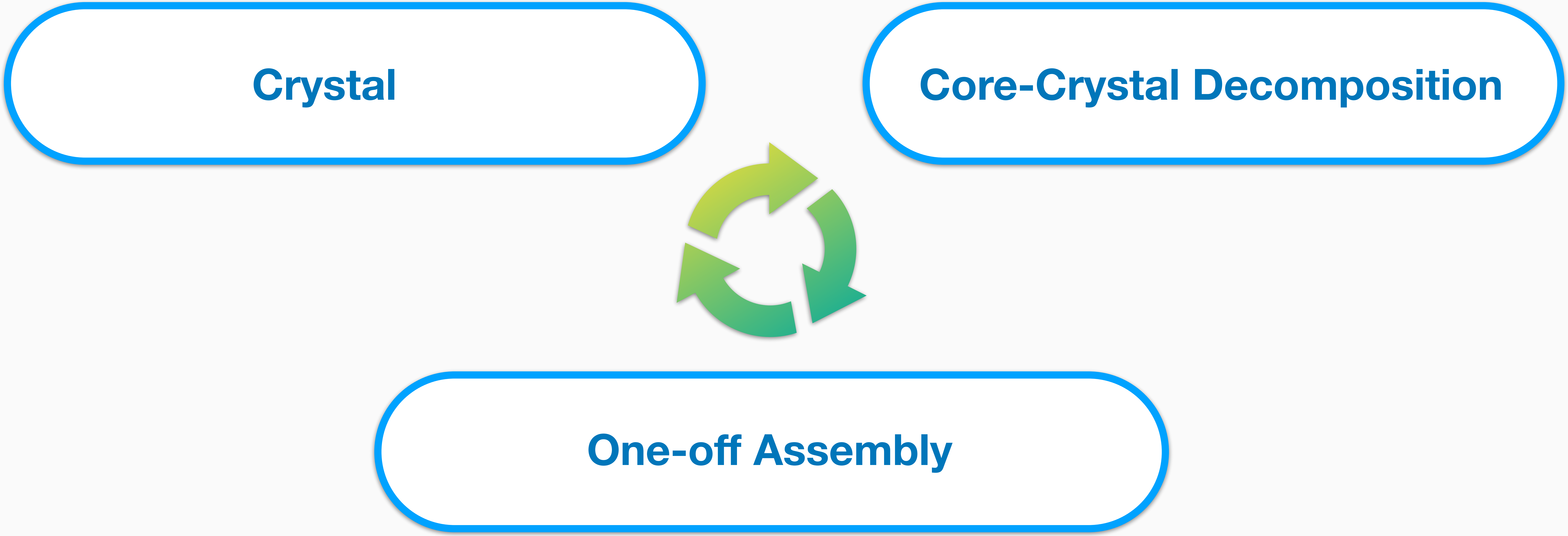
Crystal Based Computation (CBF)

Computation Solution: will the computation of subgraph matching be optimized based on the compression technique?

Avoid generating the instance set

Directly Generate Compressed Instance Set

Directly Generate Compressed Instance Set

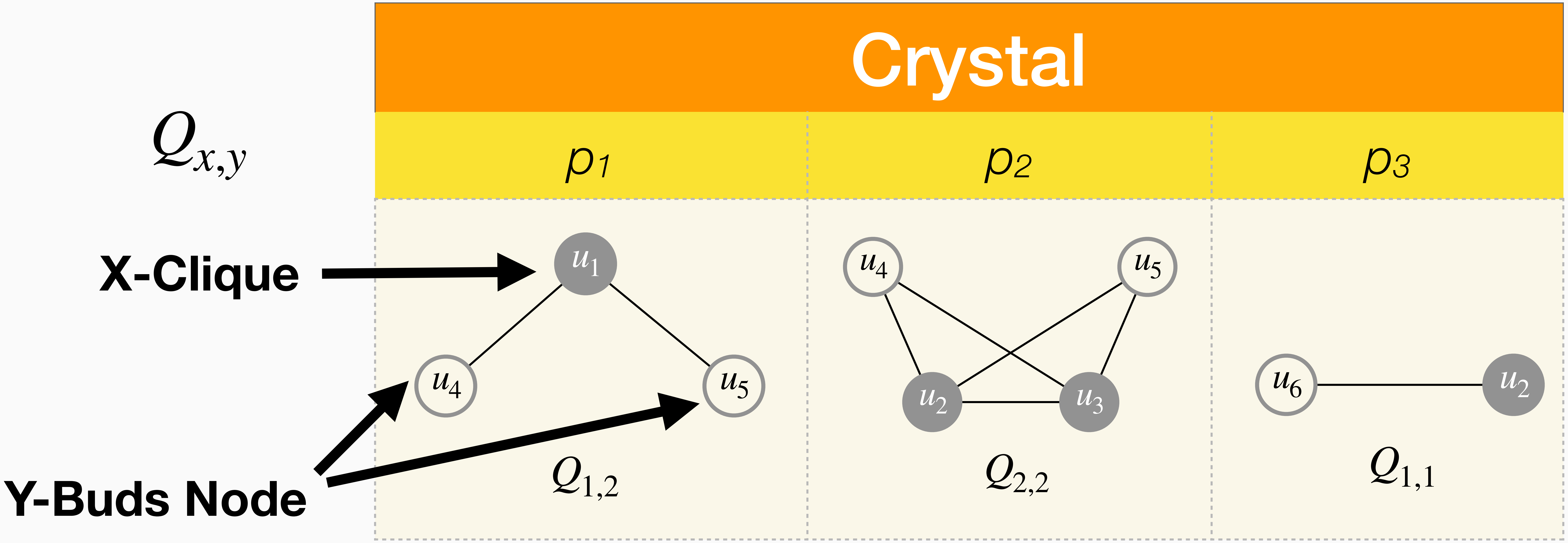


Crystal

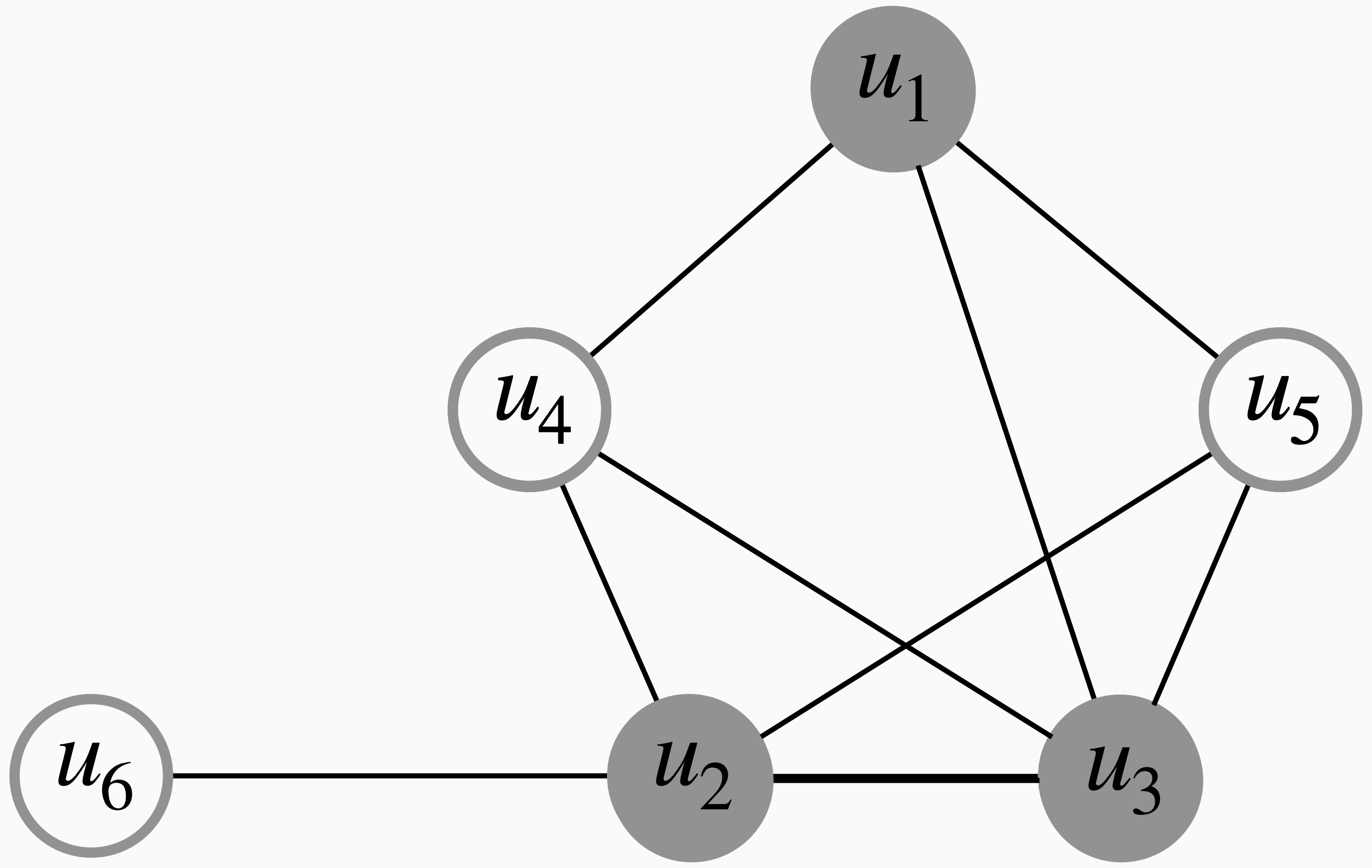
Core-Crystal Decomposition

One-off Assembly

Crystal

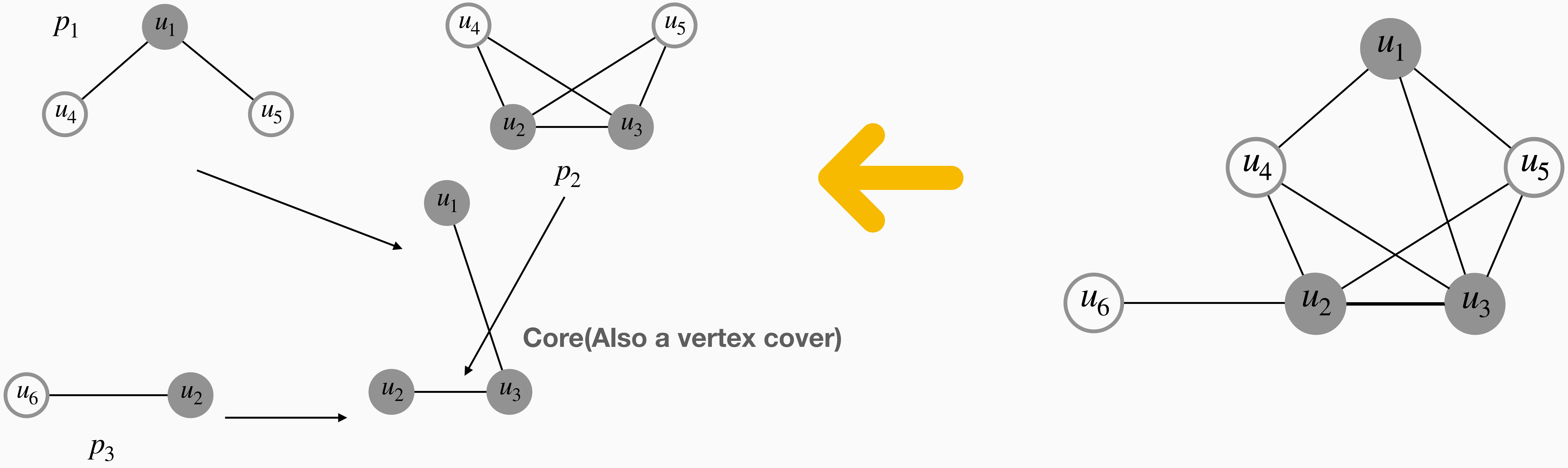

Crystal
$Q_{x,y}$
p_1
p_2
p_3
X-Clique
Y-Buds Node
u_1
u_4
u_5
$Q_{1,2}$
u_4
u_5
u_2
u_3
$Q_{2,2}$
u_6
u_2
$Q_{1,1}$

Core-Crystal Decomposition



Crystal Based Computation (CBF)

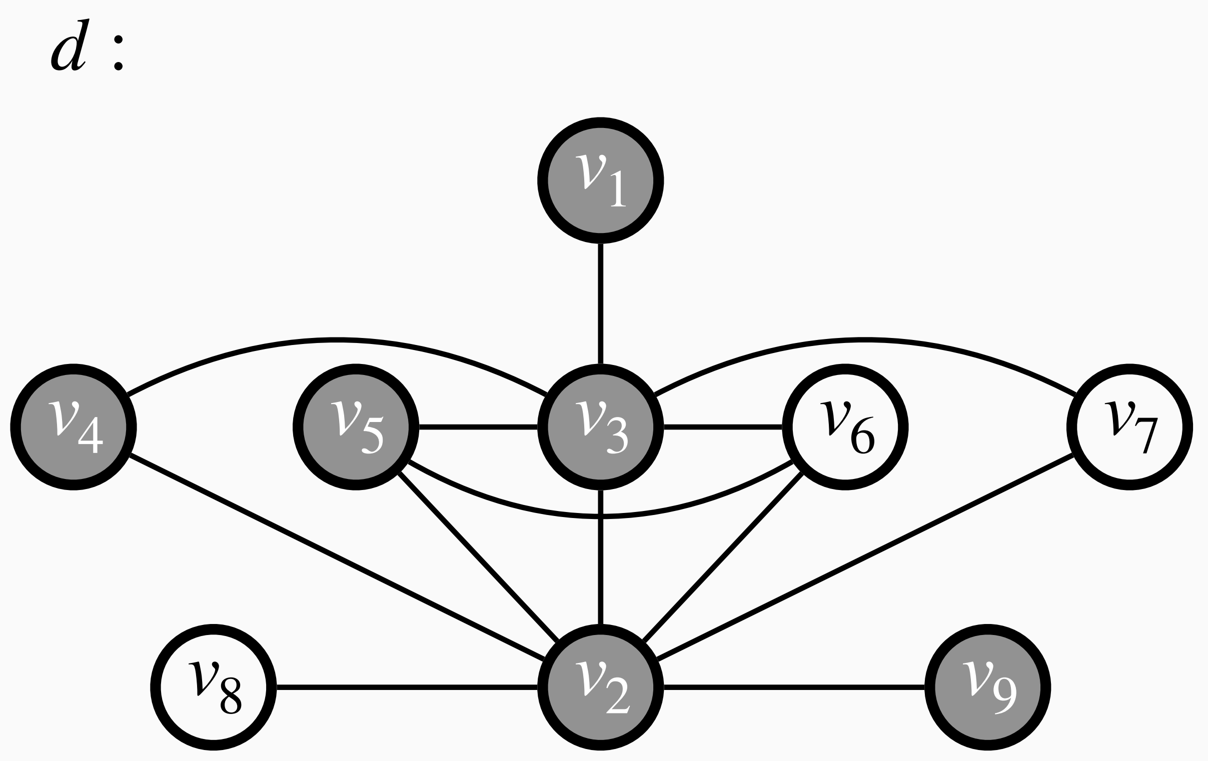
Core-Crystal Decomposition

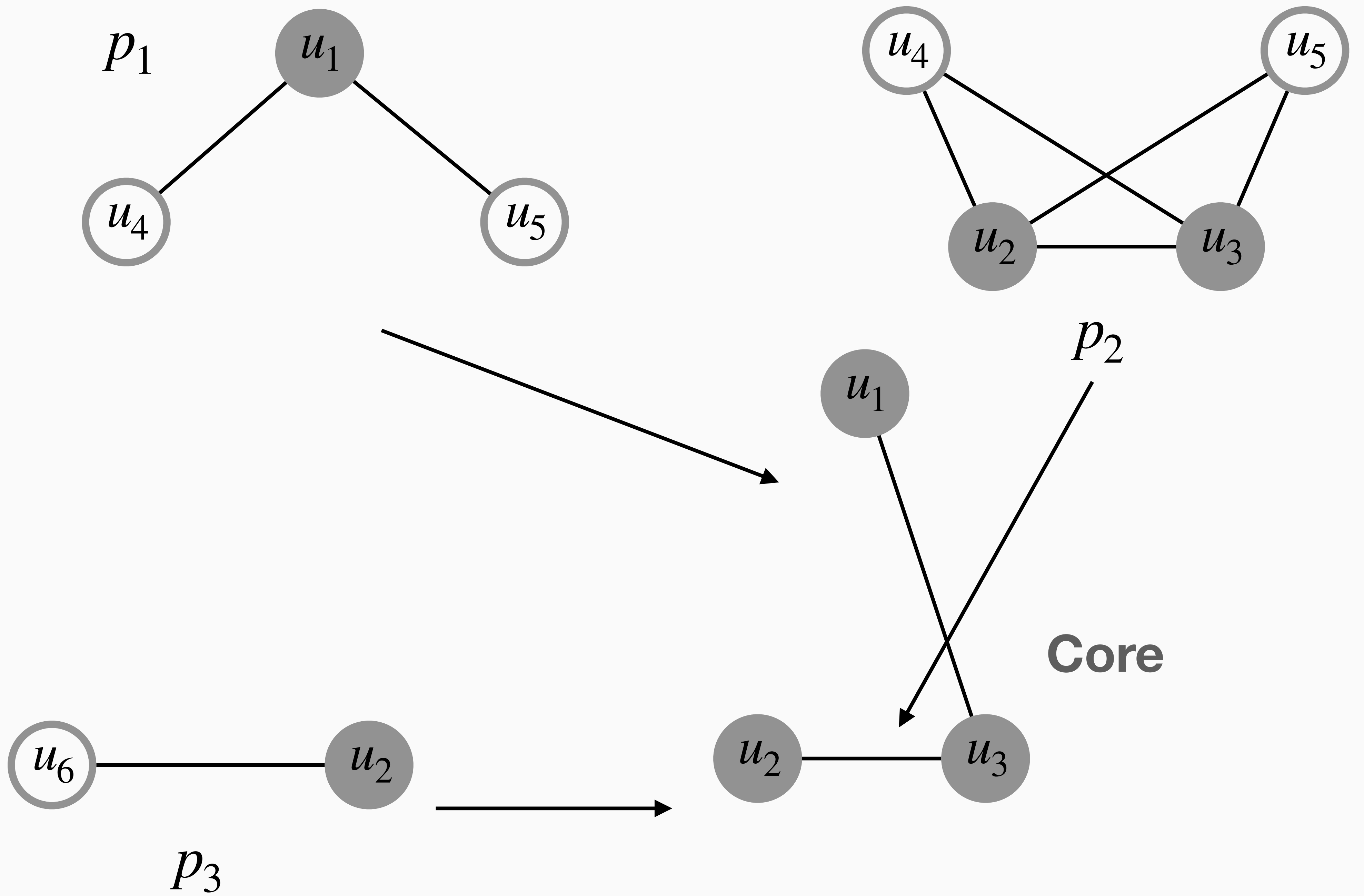


Core(Also a vertex cover)

Crystal Based Computation (CBF)

One-off Assembly



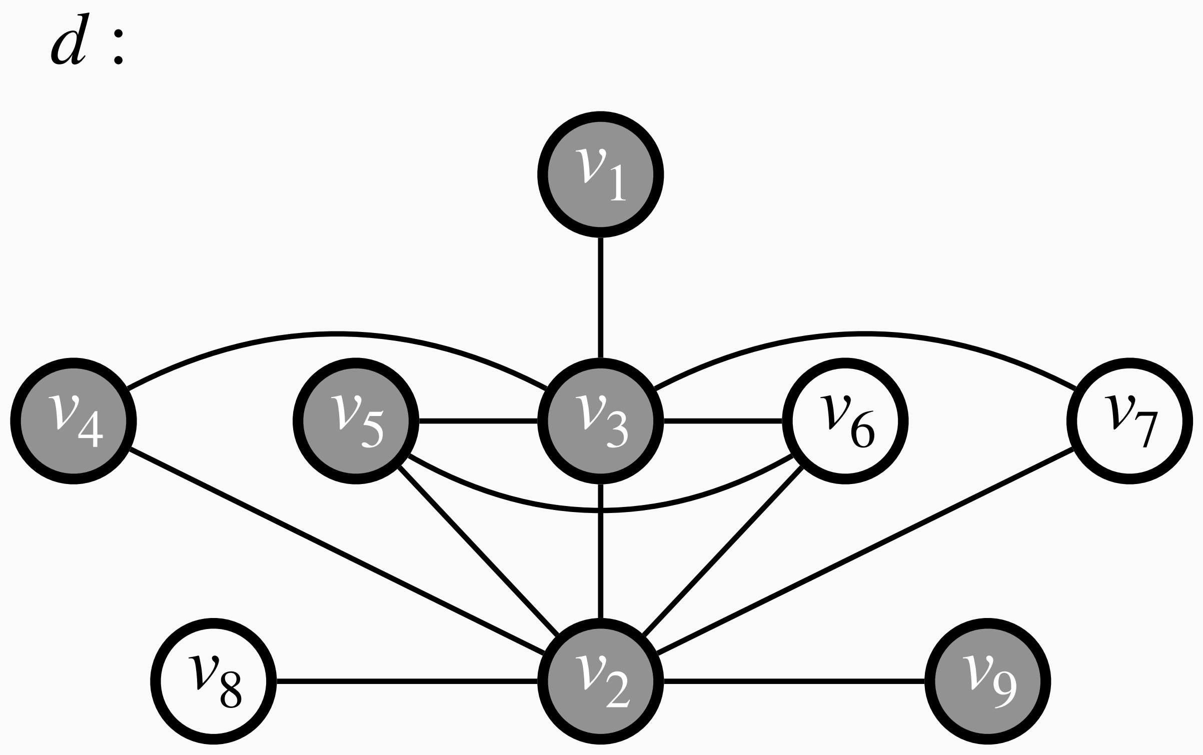


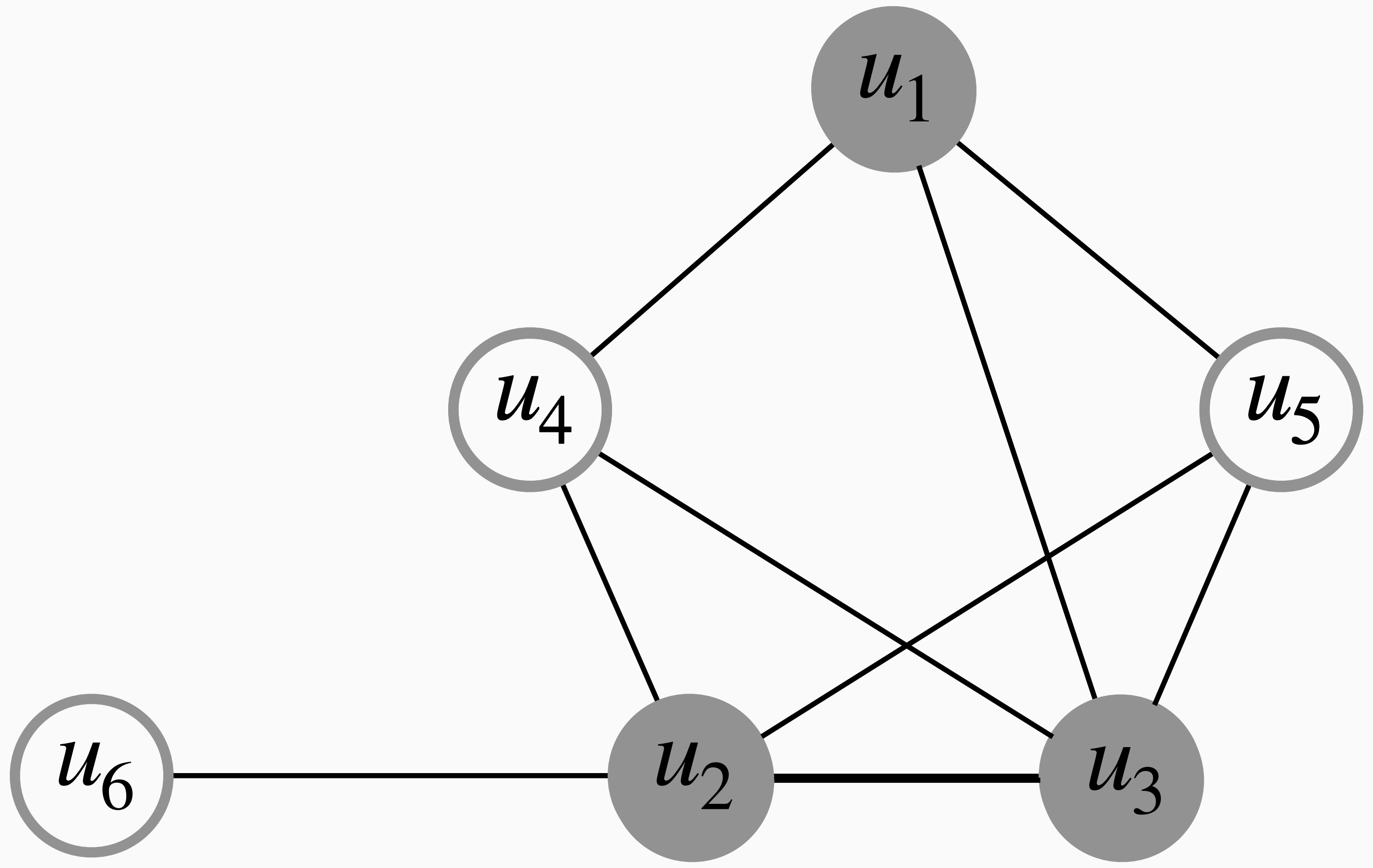
	u1	u2	u3	u4	u5	u6
Core	v1	v2	v3			
p1	v1			{v3,v4,v5,v6,v7}		
p2		v2	v3	{v4,v5,v6,v7}		
p3		v2				{v3,..,v9}
P						

Mapping for core {v1,v2,v3} and crystal

Crystal Based Computation (CBF)

One-off Assembly



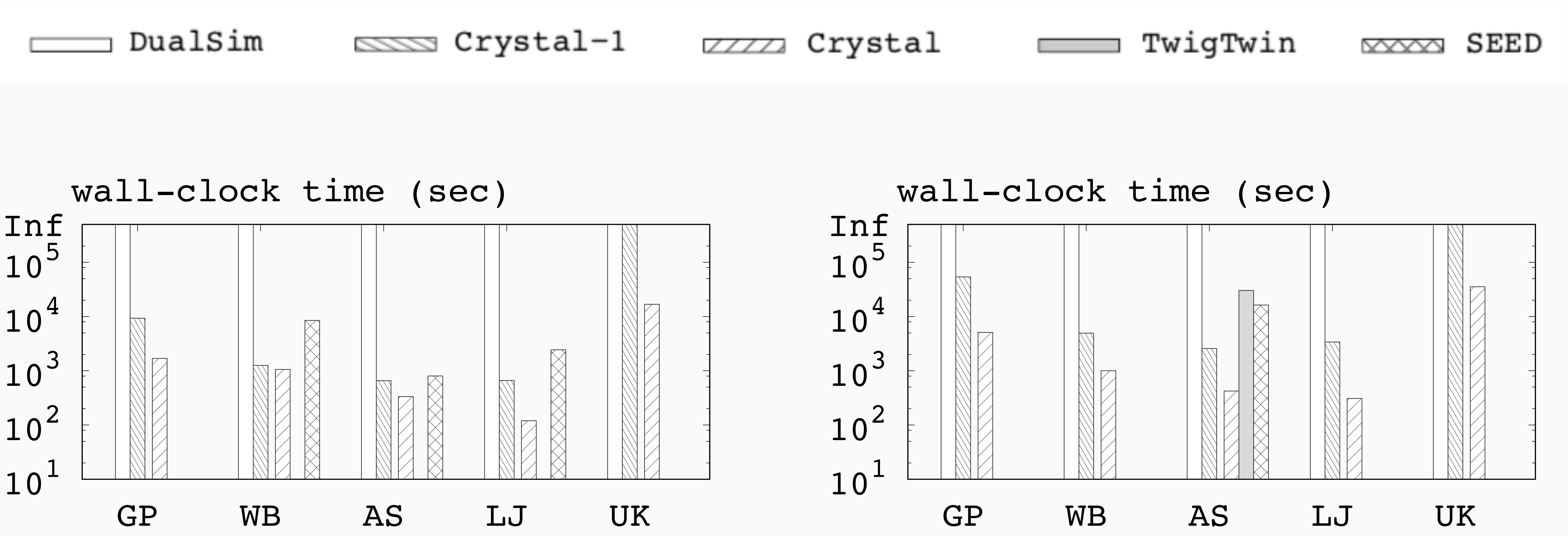


	u1	u2	u3	u4	u5	u6
Core	v1	v2	v3			
p1	v1			{v3,v4,v5,v6,v7}		
p2		v2	V3	{v4,v5,v6,v7}		
p3		v2				{v3,..,v9}
P	v1	v2	v3	{v4,v5,v6,v7}		{v3,..,v9}

Mapping for core {v1,v2,v3} and crystal

- **Crystal(Distributed) :** our methods in distributed settings
- **Crystal-1(single machine) :** our methods using one thread
- **DualSim(single-machine) :** the state-of-the-art DFS-style solution
- **TwigTwin(Distributed) :** the state-of-the-art DFS-style solution
- **SEED(Distributed) :** the state-of-the-art DFS-style solution

Experiment Computation



Q8 **Q9**

Summary

- Storage Solution For Subgraph Matching: Vertex-Cover Based Compression(VCBC)

- Direct Compressed Code Computation: Crystal-Based Framework(CBF)

Thank You !

Questions ?