		Conclusions/Questions



Enumerating all Triangulations up to Symmetry

Or: The Power of Order Rightly Used

Jörg Rambau



Lehrstuhl für Wirtschaftsmathematik

September 5–9, 2022 Workshop "Geometry meets Combinatorics" Bielefeld

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

Agenda

The Problem

Structures for Counting

Structures for Counting Subsets

New Results

Conclusions/Questions

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ ─臣 ─のへで

Agenda

The Problem

Structures for Counting

Structures for Counting Subsets

New Results

Conclusions/Questions



◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

How Many Triangulations Are There?

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ ─臣 ─のへで

Given

000

▲□▶ ▲□▶ ▲ □▶ ▲ □▶ ▲ □ ● ● ● ●

How Many Triangulations Are There?

A point configuration Given



The ProblemStructure0000000	s for Counting Structures for Counting Subsets New Results Conclusions/Questions
	How Many Triangulations Are There?
Giver	A point configuration
Question	How many triangulations does it have?
	< ロ > < 合 > < き > く き > う く や













The Problem 00●	Structures fo	or Counting	Structures for 000	Counting Subsets	New R 0000	esults 000000	Conclusio 000	ons/Questions
		Select	ed Hist	tory				
					< □ >		→ E +	E nac

The Problem 00●					Conclusions/Questions
		Select	ed History		
De	Loera 1994	Flip-base (maple-c	ed symmetric BFS in store PUNTOS)	flip-graph cor	nponent;

OOO	OOOO	or Counting	OOO	New Results	Conclusions/Questions
		Selec	ted History		
De	e Loera 1994	Flip-bas (maple-	sed symmetric BFS in -code <mark>PUNTOS</mark>)	flip-graph coi	nponent;
	R. 2000	Flip-bas simplex (oriente	sed symmetric BFS in K-by-simplex-based DI ed-matroid-based C++	flip-graph cor FS for all trian -code TOPCC	mponent; g's; 0M 0.x.x).

◆□▶ ◆昼▶ ◆臣▶ ◆臣▶ ◆□ ▶

The Problem 00●	Structures fo	or Counting	Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions	
		Selecte	ed History			
De	Loera 1994	Flip-base (maple-c	ed symmetric BFS in f ode <mark>PUNTOS</mark>)	lip-graph com	iponent;	
R	. 2000	Flip-based symmetric BFS in flip-graph component; simplex-by-simplex-based DFS for all triang's; (oriented-matroid-based C++-code TOPCOM 0.x.x).				
Santos	s 2000	The flip-	graph of triangulatior	ıs can be <mark>disc</mark> o	onnected.	

The Problem	Structures fo	or Counting	Structures for Counting Subsets	New Results 0000000000	000
		Select	ed History		
De	Loera 1994	Flip-base (maple-c	ed symmetric BFS in f code PUNTOS)	lip-graph con	nponent;
R	. 2000	Flip-based symmetric BFS in flip-graph component; simplex-by-simplex-based DFS for all triang's; (oriented-matroid-based C++-code TOPCOM 0.x.x).			
Santo	s 2000	The <mark>flip</mark> -	graph of triangulation	ns can be <mark>disc</mark>	onnected.
Ima	i et al. 2002	Flip-base Stable-se	ed reverse search for c et-based enumeration	orbits of regul of all triang's	ar triang's; 3.

OOO 0000	or Counting	OOO	New Results	000
	Selecte	ed History		
De Loera 1994	Flip-base (maple-co	d symmetric BFS in t ode <mark>PUNTOS</mark>)	flip-graph con	nponent;
R. 2000	Flip-base simplex-b (oriented	d symmetric BFS in t py-simplex-based DF -matroid-based C++-	flip-graph con S for all triang -code TOPCO	nponent; ʒ's; M 0.x.x).
Santos 2000	The flip-န	<mark>graph</mark> of triangulatio	ns can be <mark>disc</mark>	onnected.
Imai et al. 2002	Flip-base Stable-se	d reverse search for o t-based enumeration	orbits of regul of all triang's	ar triang's; s.
Jordan et al. 2018	Parallel f of sub-reg (C++-cod	lip-based reverse sea gular triang's; le MPTOPCOM).	rch for orbits	

The Problem	Structures fo	or Counting	Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		Select	ted History		
De	Loera 1994	Flip-bas (maple-	sed symmetric BFS in code PUNTOS)	flip-graph co	mponent;
F	R. 2000	Flip-bas simplex (oriente	sed symmetric BFS in -by-simplex-based DF ed-matroid-based C++	flip-graph con S for all trian -code TOPCC	mponent; g's;)M 0.x.x).
Santo	os 2000	The flip	- <mark>graph</mark> of triangulatic	ons can be <mark>dis</mark> o	connected.
Ima	i et al. 2002	Flip-bas Stable-s	sed reverse search for set-based enumeration	orbits of regu 1 of all triang'	lar triang's; s.
Jordaı	n et al. 2018	Parallel of sub-r (C++-co	flip-based reverse sea regular triang's; ode MPTOPCOM).	arch for orbits	
F	New: R. 2022	Parallel for all tr (new C-	symmetric lexicograp riang's ++-code TOPCOM 1.x	ohic subset rev .x),	/erse search



The Problem 000	Structures for Counting 0●00	Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions 000
	Reve	rse Search (RS)		
			 < □ > < □ > < □ 	ま → ま ♪ ま りへ(~

Goal Enumerate the nodes (= objects) of a graph (V, E) with

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ のへぐ

Goal

Enumerate the nodes (= objects) of a graph (V, E) with

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

an objective function on V with unique opt v_{opt} ;

Goal

Enumerate the nodes (= objects) of a graph (V, E) with

- an objective function on V with unique opt v_{opt} ;
- a pivot function choosing a better neighbor on $V \setminus \{v_{opt}\}$.

Goal Enumerate the nodes (= objects) of a graph (V, E) with

- an objective function on V with unique opt v_{opt} ;
- a pivot function choosing a better neighbor on $V \setminus \{v_{opt}\}$.

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

Method Reverse Search (RS) [Avis & Fukuda 1996]:

Goal Enumerate the nodes (= objects) of a graph (V, E) with

- an objective function on V with unique opt v_{opt} ;
- a pivot function choosing a better neighbor on $V \setminus \{v_{opt}\}$.

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

Method Reverse Search (RS) [Avis & Fukuda 1996]:

- **Goal** Enumerate the nodes (= objects) of a graph (V, E) with
 - an objective function on V with unique opt v_{opt} ;
 - a pivot function choosing a better neighbor on $V \setminus \{v_{opt}\}$.

Method Reverse Search (RS) [Avis & Fukuda 1996]:

generate an arbitrary object



a pivot function choosing a better neighbor on $V \setminus \{v_{opt}\}$.

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

Method Reverse Search (RS) [Avis & Fukuda 1996]:

generate an arbitrary object

pivot to the optimum object











	s for Counting	000	0000000000	000		
	Rever	se Search (RS)				
Goa	Enumer	ate the nodes (= obje	cts) of a graph	(V, E) with		
•	an objec	ctive function on V w	vith unique op	t v _{opt} ;		
•	a pivot t	a pivot function choosing a better neighbor on $V \setminus \{v_{opt}\}$.				
Method	Reverse	Search (RS) [Avis &	Fukuda 1996]:			
•	generat	e an arbitrary object				
•	pivot to	the optimum object				
•	return F	ReverseSearch(optimu	um object)			
Subroutine	Reverse	Search(object):				
•	increase	e counter				
•	for all n	eighbors of object do)			
	if neigl	hbor pivots to object				
	increas	se counter by ReverseSe	earch(object)			

▲□▶▲圖▶▲≣▶▲≣▶ ≣ のQで

000	0000	or Counting	000	anting subsets	000000000000000000000000000000000000000	000	
		Rever	rse Search	n (RS)			
	Goal	Enume	rate the nod	es (= objec	ts) of a graph	(V, E) with	ı
	►	an obje	ctive function	on on V wit	th unique opt	$v_{\rm opt};$	
	►	a pivot	function cho	oosing a be	tter neighbor	on $V \setminus \{v_{\text{op}}\}$	$_{\rm pt}\}.$
٨	Aethod	Reverse	e Search <mark>(RS</mark>) [Avis & Fi	ukuda 1996]:		
		generat	te an arbitra	ry object			
	•	pivot to	o the optimu	m object			
	•	return	ReverseSear	ch(optimur	n object)		
Subi	routine	Reverse	eSearch(obje	ct):			
		increas	e counter				
		for all r	neighbors of	object do			
	•	if neig increa	hbor pivots to se counter by	o object • ReverseSea	rch(object)		
	•	return	counter.		(1)	▶ < ≣ > = -	୬୯୯
Structures for Counting	Structures for Counting Subsets	Conclusions/Questions					
-------------------------	---------------------------------	-----------------------					
0000							

◆□ ▶ ◆□ ▶ ◆ □ ▶ ◆ □ ▶ ● □ ● ● ● ●

RS Example

New Results

・ロト ・個ト ・ヨト ・ヨト ・ヨー

Conclusions/Questions

RS Example



New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example




New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





New Results

Conclusions/Questions

RS Example





	Structures for Counting		Conclusions/Questi 000

Reverse Search on Orbits

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ - □ - のへぐ

New Results

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ ─臣 ─のへで

Reverse Search on Orbits

Canonical Representatives Function "G-Orbits \rightarrow Elements", e.g., \mapsto \mapsto



▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()



< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>



▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()





Structures for Counting	Structures for Counting Subsets	Conclusions/Questions/
	000	

Representation of Objects as Subsets

Representation of Objects as Subsets

Observation

Many objects have a representation as subsets S of $\{1, \ldots, n\}$.

(日)

The Problem 000	Structures fo		Structures for Counting Subsets ○●○	New Results 0000000000	Conclusions/Questions
		Repre	sentation of Obj	ects as Su	bsets
Observation		Many objects have a representation as subsets S of $\{1, \ldots, n\}$.			
Idea		Build objects by lex-extension.			

◆□▶ ◆□▶ ◆三▶ ◆三▶ ▲□▶ ▲□▶

The Problem	Structures fo	or Counting Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions	
		Representation of Obj	ects as Su	bsets	
Observation Idea Gain		Many objects have a representation as subsets S of $\{1, \ldots, n\}$.			
		Build objects by lex-extension	1.		
		Subset Reverse Search (SRS):			

◆□▶ ◆昼▶ ◆臣▶ ◆臣▶ ◆□ ▶

The Problem	Structures fo	or Counting Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions	
		Representation of Obj	ects as Su	bsets	
Observation Idea Gain		Many objects have a representation as subsets S of $\{1, \ldots, n\}$.			
		Build objects by lex-extension.			
		Subset Reverse Search (SRS):			

◆□▶ ◆昼▶ ◆臣▶ ◆臣▶ ◆□ ▶
The Problem 000	Structures fo		Structures for Counting Subsets ○●○	New Results 0000000000	Conclusions/Questions	
		Repre	sentation of Obj	ects as Sul	osets	
Obser	vation	Many objects have a representation as subsets S of $\{1, \ldots, n\}$.				
	Idea	Build objects by lex-extension.				
	Gain Subset Reverse Search (SRS):					
 Lex-order is an objective with easy opt 			∎easy opt Ø			

The Problem 000	Structures fo		Structures for Counting Subsets ○●○	New Results 0000000000	Conclusions/Questions	
		Repres	entation of Obj	ects as Sub	osets	
Observ	vation	Many obj <i>S</i> of {1,	ects have a represen ., <i>n</i> }.	tation as subs	sets	
	Idea	Build objects by lex-extension.				
	Gain	Subset Re	everse Search <mark>(SRS)</mark> :			
		Lex-order is an objective with easy opt \emptyset				
	•	Removing max-element view easily invertible pivot			pivot	

The Problem 000	Structures fo	or Counting	Structures for Counting Subsets ○●○	New Results 0000000000	Conclusions/Questions	
		Repres	sentation of Obj	ects as Sub	osets	
Observ	ation	Many of <i>S</i> of {1, .	<pre>ojects have a represer , n}.</pre>	ntation as subs	sets	
	Idea	Build objects by lex-extension.				
	Gain	Subset Reverse Search (SRS):				
		Lex-order is an objective with easy opt \emptyset				
	•	Removing max-element ~~> easily invertible pivot				
		\Rightarrow SRS ϵ	enumerates subsets.			

The Problem 000	Structures fo		Structures for Counting Subsets 0●0	New Results 0000000000	Conclusions/Questions 000	
		Repre	esentation of Obj	ects as Sul	osets	
Obser	vation	Many objects have a representation as subsets S of $\{1, \ldots, n\}$.				
	Idea	Build objects by lex-extension.				
	Gain	Subset Reverse Search (SRS):				
		Lex-order is an objective with easy opt Ø				
		Removing max-element ~~> easily invertible pivot				
\Rightarrow SRS enumerates subsets.						
(Crucial	Need to	o recognize <mark>complete</mark> d	objects.		

The Problem Stru 000 00	uctures fo 00		Structures for Count	ing Subsets	New Results 0000000000	Conclusions/Questions 000
		Repres	sentation	of Obje	cts as Sub	osets
Observat	ion	Many of <i>S</i> of {1, .	ojects have a , <i>n</i> }.	represent	ation as subs	sets
le	dea	Build ob	jects by <mark>lex-e</mark>	extension.		
G	Gain Subset Reverse Search (SRS):					
		Lex-orde	er is an objec	tive with o	easy opt Ø	
		Removing max-element ~-> easily invertible pivot				
		\Rightarrow SRS ϵ	enumerates s	ubsets.		
Cru	cial	Need to	recognize <mark>co</mark>	m <mark>plete</mark> ob	jects.	
Overh	ead	SRS take	es additional	time, sinc	e	

The Problem Str	ructures fo 200	r Counting	Structures for Counting Su ○●○	bsets	New Results 0000000000	Conclusions/Questions
		Repre	sentation of	Obje	cts as Sul	osets
Observa	tion	Many ol <i>S</i> of {1, .	bjects have a rep , n}.	resent	ation as subs	sets
I	Idea	Build objects by lex-extension.				
(Gain	Subset Reverse Search (SRS):				
	•	Lex-order is an objective with easy opt Ø				
		Removing max-element ~-> easily invertible pivot				
		\Rightarrow SRS of	enumerates subs	ets.		
Cru	icial	Need to	recognize comp	<mark>lete</mark> ob	jects.	
Overh	lead	SRS tak	es additional tim	ie, sinc	e	
		all lex-le	ading subsets of	object	ts are travers	sed

The Problem	Structures fo	or Counting	Structures for Counting Subsets ○●○	New Results 0000000000	Conclusions/Questions	
		Repre	sentation of Ob	ojects as Su	bsets	
Observ	vation	Many o <i>S</i> of {1,	bjects have a represe , n}.	entation as sub	sets	
	Idea	Build of	ojects by <mark>lex-extensions</mark>	on.		
	Gain	Subset I	Reverse Search <mark>(SRS</mark>)):		
► Lex-order is an objective with easy opt Ø						
		Removing max-element variable pivot				
		\Rightarrow SRS	enumerates subsets.			
C	rucial	Need to	recognize <mark>complete</mark>	objects.		
Ove	rhead	SRS tak	es additional time, s	ince		
		all lex-le	eading subsets of obj	jects are traver	sed	
	•	there m	ay be dead-ends w.r.	.t. lex-extensior	ı	

The Problem 000	Structures fo	or Counting	Structures for Counting Subsets ○●○	New Results 0000000000	Conclusions/Questions
		Repre	sentation of Ob	jects as Su	bsets
Observ	ation	Many o <i>S</i> of {1, 1	bjects have a represe , n}.	entation as sub	osets
	Idea	Build of	ojects by <mark>lex-extensic</mark>	on.	
	Gain	Subset I	Reverse Search <mark>(SRS)</mark>	:	
	•	Lex-order is an objective with easy opt Ø			
	•	Removi	ng max-element 🛶 e	easily invertible	<mark>e</mark> pivot
		\Rightarrow SRS	enumerates subsets.		
С	rucial	Need to	recognize complete	objects.	
Over	rhead	SRS tak	es additional time, si	ince	
	•	all lex-le	eading subsets of obj	ects are traver	sed
	•	there m	ay be dead-ends w.r.	t. lex-extension	n
	•	contain	ment in an object ma	ay be difficult t	to tell early
				Image: A matrix and a matrix	E> < E> E • େ २

	Structures for Counting Subsets 00●	Conclusions/Qu 000

Generic Algorithm: Symmetric LSRS

◆□ ▶ < @ ▶ < E ▶ < E ▶ E • 9 < @</p>

(日)

Generic Algorithm: Symmetric LSRS

Observation

Subset *S* lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit.

The Problem 000	Structures fo	or Counting Structures for Counting Subse	ts New Results 0000000000	Conclusions/Questions		
		Generic Algorithm:	Symmetric L	.SRS		
Obser	rvation	Subset S lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit.				
Punch Line ca		canonical = lex-min \Longrightarrow canonicals connected				

The Problem	OOOO	or Counting Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions		
		Generic Algorithm: Sy	mmetric l	SRS		
Obser	vation	Subset S lex-min in its orbit \implies S \ max S lex-min in its o	rbit.			
Punch Line		canonical = lex-min \Longrightarrow canonicals connected				
Gain		Symmetric Lexicographic Sub (symLSRS) [equivalent: Pech	oset Reverse S & Reichard 2	earch 009]:		

The Problem	OOOO	or Counting Structures for Counting Subsets	New Results	OOO		
		Generic Algorithm: Sy	mmetric L	.SRS		
Observation		Subset S lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit.				
Punch Line		canonical = lex-min \Longrightarrow canonicals connected				
Gain		Symmetric Lexicographic Sub (symLSRS) [equivalent: Pech	oset Reverse S & Reichard 24	earch 009]:		

000 0000	s for Counting Structures for Counting Subsets	00000000000000000000000000000000000000	Conclusions/Questions
	Generic Algorithm: Sy	ymmetric L	_SRS
Observation	Subset S lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit	orbit.	
Punch Line	canonical = lex-min \Longrightarrow cano	onicals connec	ted
Gaiı	Symmetric Lexicographic Su (symLSRS) [equivalent: Pech Input: a subset S	bset Reverse S 1 & Reichard 20	earch 009]:

The Problem	Structures fo	r Counting Structures for Counting Subs	ets New Results	Conclusions/Questions
		Generic Algorithm:	Symmetric I	_SRS
Obser	vation	Subset S lex-min in its orl $\implies S \setminus \max S$ lex-min in i	oit ts orbit.	
Punc	h Line	canonical = lex-min \Longrightarrow c	anonicals connec	ted
	Gain	Symmetric Lexicographic (symLSRS) [equivalent: P Input: a subset S	Subset Reverse S ech & Reichard 2	Search 1009]:
	•	if <i>S</i> not lex-extendable to	an object, return	0

The Problem	Structures to	or Counting Structures for Counting Subsets	New Results	Conclusions/Questions
		Generic Algorithm: S	Symmetric L	SRS
Obser	vation	Subset S lex-min in its orbit \implies S \ max S lex-min in its	orbit.	
Punc	h Line	canonical = lex-min \Longrightarrow car	ionicals connec	ted
	Gain	Symmetric Lexicographic S (symLSRS) [equivalent: Pec Input: a subset S	ubset Reverse S h & Reichard 20	earch 009]:
	•	if S not lex-extendable to an	<mark>ו object</mark> , return	0

◆□▶ ◆昼▶ ◆臣▶ ◆臣▶ ◆□ ▶

The Problem Structur	s for Counting Structures for Counting Subsets New Results Conclusions/Questions 00● 000000000 000
	Generic Algorithm: Symmetric LSRS
Observatio	Subset S lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit.
Punch Lin	canonical = lex-min \implies canonicals connected
Gai	 Symmetric Lexicographic Subset Reverse Search (symLSRS) [equivalent: Pech & Reichard 2009]: Input: a subset S
1	if <i>S</i> not lex-extendable to an object, return 0
1	if <i>S</i> not lex-min in its orbit, return 0

The Problem 000	Structures fo	or Counting	Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		Gene	ric Algorithm: S	Symmetric L	.SRS
Obser	vation	$\begin{array}{l} Subset \\ \Longrightarrow S \setminus \end{array}$	<i>S</i> lex-min in its orbit max <i>S</i> lex-min in its	t s orbit.	
Punc	h Line	canonic	$cal = lex-min \Longrightarrow can$	nonicals connect	ted
	Gain	Symme <mark>(symLS</mark> Input: a	etric Lexicographic S <mark>RS)</mark> [equivalent: Pec a subset <i>S</i>	ubset Reverse S ch & Reichard 20	earch)09]:
		if <i>S</i> not	lex-extendable to a	<mark>n object</mark> , return	0
	•	if <i>S</i> not	lex-min in its orbit,	return 0	

000 0000	s for Counting Subsets New Results Conclusions/Questions
	Generic Algorithm: Symmetric LSRS
Observation	Subset S lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit.
Punch Line	canonical = lex-min \implies canonicals connected
Gain	Symmetric Lexicographic Subset Reverse Search (symLSRS) [equivalent: Pech & Reichard 2009]: Input: a subset <i>S</i>
►	if S not lex-extendable to an object, return 0
•	if S not lex-min in its orbit, return 0
•	if <i>S</i> is a complete object, return 1

◆□▶ ◆昼▶ ◆臣▶ ◆臣▶ ◆□ ▶

000	0000	or Counting	000	00000000000000000000000000000000000000	000
		Gener	ric Algorithm:	Symmetric L	.SRS
Obser	vation	Subset $S \rightarrow S \setminus S$	S lex-min in its orbi max S lex-min in it	t s orbit.	
Punc	h Line	canonic	al = lex-min \Longrightarrow ca	nonicals connec	ted
	Gain	Symmet <mark>(symLSI</mark> Input: a	tric Lexicographic S <mark>RS)</mark> [equivalent: Pe subset <i>S</i>	Subset Reverse S ch & Reichard 20	earch 009]:
		if <i>S</i> not	lex-extendable to a	<mark>n object</mark> , return	0
	•	if S not	lex-min in its orbit	, return 0	
		if <i>S</i> is a	complete object, re	turn 1	
	•	for <i>i</i> fro	m max <i>S</i> + 1, , <i>n</i> :		

◆□▶ ◆昼▶ ◆臣▶ ◆臣▶ ◆□ ▶

000 0000	oo 000 00000000 000
	Generic Algorithm: Symmetric LSRS
Observatior	Subset S lex-min in its orbit $\implies S \setminus \max S$ lex-min in its orbit.
Punch Line	canonical = lex-min \implies canonicals connected
Gair	Symmetric Lexicographic Subset Reverse Search (symLSRS) [equivalent: Pech & Reichard 2009]: Input: a subset <i>S</i>
•	if S not lex-extendable to an object, return 0
•	if S not lex-min in its orbit, return 0
•	if S is a complete object, return 1
•	for <i>i</i> from max $S + 1, \ldots, n$:
	increase counter by symLSRS($S \cup \{i\}$)

▲□▶▲圖▶▲≣▶▲≣▶ ≣ のQで

000	0000	OO•	0000000000	000	
		Generic Algorithm: S	ymmetric L	.SRS	
Obser	vation	Subset S lex-min in its orbit \implies S \ max S lex-min in its	orbit.		
Punc	h Line	canonical = lex-min \Longrightarrow can	onicals connect	ted	
	Gain	Symmetric Lexicographic Su (symLSRS) [equivalent: Peck Input: a subset <i>S</i>	ıbset Reverse S 1 & Reichard 20	earch 009]:	
	•	if S not lex-extendable to an	object, return	0	
	•	if S not lex-min in its orbit,	return 0		
	•	if S is a complete object, ret	urn 1		
	•	for <i>i</i> from max $S + 1, \ldots, n$:			
	•	increase counter by symLSRS	$S(S \cup \{i\})$		
	•	return counter			
				→ < E > E	590

000	0000	or Counting	000	nting subsets	000000000000000000000000000000000000000	000
		Gene	ric Algori	thm: Sy	mmetric L	.SRS
Observ	vation	$\begin{array}{l} Subset \\ \Longrightarrow S \setminus \end{array}$	S lex-min in max S lex-m	its orbit in in its or	bit.	
Punc	h Line	canonic	al = lex-min	\implies canon	icals connec	ted
	Gain	Symme <mark>(symLS</mark> Input: a	tric Lexicogı <mark>RS)</mark> [equival 1 subset <i>S</i>	raphic Subs ent: Pech ک	set Reverse S & Reichard 20	earch 009]:
		if <i>S</i> not	lex-extenda	ble to an o	bject, return	0
	•	if <i>S</i> not	lex-min in i	ts orbit, ret	urn 0	
	•	if <i>S</i> is a	complete of	oject, retur	n 1	
	•	for <i>i</i> fro	m max $S + 1$,, <i>n</i> :		
	•	increa	se counter by	symLSRS(S	$S \cup \{i\}$)	
	•	return o	counter			
		\rightarrow sym	LSRS(Ø) lex-	enumerate	s all orbit-le>	k-min objects
						▶ ▲ 臣 ▶ □ 目 → の �





| ◆ □ ▶ ◆ □ ▶ ◆ □ ▶ ● □ ● ● ● ● ●



◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ のへぐ





< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>



< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

Structures for Counting	

Structures for Counting Subsets

New Results

Conclusions/Questions

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

Ingredient I: Critical Elements



Ingredient I: Critical Elements

Theorem

Let *S* be a subset that is lex-min in its orbit. Then for all $i \in \{\max S + 1, ..., n\}$ we have:

 $S \cup \{i\}$ is **not** lex-min in its orbit

 \iff

there is a $\pi \in G$ with:





there is a $\pi \in G$ with:

- $\operatorname{crit}_{S}(\pi) = \infty$ and $\pi(i) < \max S$, or
- $\operatorname{crit}_{S}(\pi) \neq \infty$ and $\pi(i) < \operatorname{crit}_{S}(\pi)$, or

Ingredient I: Critical Elements

Theorem

Let S be a subset that is lex-min in its orbit. Then for all $i \in \{\max S + 1, ..., n\}$ we have: $S \cup \{i\}$ is not lex-min in its orbit \longleftrightarrow there is a $\pi \in G$ with:

- $\operatorname{crit}_{S}(\pi) = \infty$ and $\pi(i) < \max S$, or
- $\operatorname{crit}_{S}(\pi) \neq \infty$ and $\pi(i) < \operatorname{crit}_{S}(\pi)$, or
- $\operatorname{crit}_{S}(\pi) = \pi(i) \text{ and } \operatorname{crit}_{S \cup \{i\}}(\pi) \in \pi(S \cup \{i\}).$

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>



|▲□▶ ▲圖▶ ▲≣▶ ▲≣▶ | ≣ | 釣��
New Results 0000000000 Ingredient I: Critical Elements Theorem Let S be a subset that is lex-min in its orbit. Then for all $i \in \{\max S + 1, \dots, n\}$ we have: $S \cup \{i\}$ is not lex-min in its orbit there is a $\pi \in G$ with: $\operatorname{crit}_{S}(\pi) = \infty$ and $\pi(i) < \max S$, or $\operatorname{crit}_{S}(\pi) \neq \infty$ and $\pi(i) < \operatorname{crit}_{S}(\pi)$, or $\operatorname{crit}_{S}(\pi) = \pi(i)$ and $\operatorname{crit}_{S \cup \{i\}}(\pi) \in \pi(S \cup \{i\})$. Gain $\pi(S \cup \{i\})$ only needed if $\operatorname{crit}_{S}(\pi) = \pi(i)$ \rightarrow roughly $\frac{1}{n}$ of the cases (amortized)

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

	Structures for Counting Subsets	New Results	Conclusions/Question
		0000000000	

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ のへぐ

Triangulations as Integer-Subsets

Representation

For a point configuration of *n* points in rank *r*:

Triangulations as Integer-Subsets

Representation

For a point configuration of *n* points in rank *r*:

S: the set of *r*-simplices, lex-ordered

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

 $n_s = |S|$: no. of simplices

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

- $n_s = |S|$: no. of simplices
- $n_f = |\mathcal{F}|$: no. of interior facets of simplices

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

- $n_s = |S|$: no. of simplices
- $n_f = |\mathcal{F}|$: no. of interior facets of simplices
- simp : $\{1, \ldots, n_s\} \stackrel{\sim}{\leftrightarrow} S$: s-index (order-preserving)

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered
- $n_s = |S|$: no. of simplices
- $n_f = |\mathcal{F}|$: no. of interior facets of simplices
- simp : $\{1, \ldots, n_s\} \stackrel{\sim}{\leftrightarrow} S$: s-index (order-preserving)
- ▶ facet : $\{1, ..., n_f\} \stackrel{\sim}{\leftrightarrow} \mathcal{F}$: f-index (order-preserving)

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered
- $n_s = |S|$: no. of simplices
- $n_f = |\mathcal{F}|$: no. of interior facets of simplices
- simp : $\{1, \ldots, n_s\} \stackrel{\sim}{\leftrightarrow} S$: s-index (order-preserving)
- ▶ facet : $\{1, ..., n_f\} \stackrel{\sim}{\leftrightarrow} \mathcal{F}$: f-index (order-preserving)

▲□▶▲□▶▲□▶▲□▶ □ ● ●

• $\mathcal{T} = {\operatorname{simp}(s) : s \in T}$ for $T \subseteq {1, \ldots, n_s}$

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
- \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered
- $n_s = |S|$: no. of simplices
- $n_f = |\mathcal{F}|$: no. of interior facets of simplices
- simp : $\{1, \ldots, n_s\} \stackrel{\sim}{\leftrightarrow} S$: s-index (order-preserving)
- ▶ facet : $\{1, ..., n_f\} \stackrel{\sim}{\leftrightarrow} \mathcal{F}$: f-index (order-preserving)

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

- $\mathcal{T} = {\operatorname{simp}(s) : s \in T}$ for $T \subseteq {1, \ldots, n_s}$
- $T = \{s\text{-index}(S) : S \in \mathcal{T}\} \text{ for } \mathcal{T} \subseteq S$

Representation

For a point configuration of *n* points in rank *r*:

- S: the set of *r*-simplices, lex-ordered
 - \mathcal{F} : the set of interior facets of *r* simplices, lex-ordered
- $n_s = |S|$: no. of simplices
- $n_f = |\mathcal{F}|$: no. of interior facets of simplices
- simp : $\{1, \ldots, n_s\} \stackrel{\sim}{\leftrightarrow} S$: s-index (order-preserving)
- ▶ facet : $\{1, ..., n_f\} \stackrel{\sim}{\leftrightarrow} \mathcal{F}$: f-index (order-preserving)
- $\mathcal{T} = {\operatorname{simp}(s) : s \in T}$ for $T \subseteq {1, \ldots, n_s}$
- $T = \{s\text{-index}(S) : S \in \mathcal{T}\} \text{ for } \mathcal{T} \subseteq S$

Convention

All $s \in \{1, ..., n_s\}$ and $f \in \{1, ..., n_f\}$ are called simplices and facets, resp. All $T \subseteq \{1, ..., n_s\}$ with pairwise proper intersections are called partial triangulations.



- ◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 - のへぐ

tructures for Counting Subsets

New Results

Conclusions/Questions

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

Is a Subset Not Lex-Extendable?

Extendability Check [Ruppert & Seidel 1992]: Extendability of partial triangulations is NP-complete.

Is a Subset Not Lex-Extendable?

Extendability Check Observation [Ruppert & Seidel 1992]:

Extendability of partial triangulations is NP-complete.

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

Each interior facet must be covered by additional simplices to complete a triangulation.

The Problem 000	Structures fo	or Counting	Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		ls a S	ubset Not Lex-Ex	tendable?	
Extend	dability Check	[Ruppe Extend	ert & Seidel 1992]: ability of partial triang	ulations is N	P-complete.
Observation Eac sim		Each ir simplic	Each interior facet must be covered by additional simplices to complete a triangulation.		
Global Data Prepro		cess for each simplex s			

The Problem	Structures fo		Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		ls a S	ubset Not Lex-Ex	tendable?	
Extend	dability Check	[Ruppe Extend	ert & Seidel 1992]: ability of partial triang	gulations is NI	P-complete.
Observation Each i simpli		Each ir simplic	nterior facet must be covered by additional ces to complete a triangulation.		
Glob	al Data	Prepro	cess for each simplex s		
	•	its inte	rior facets $\rightsquigarrow \mathcal{F}(s)$		

The Problem 000	Structures fo	or Counting Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		Is a Subset Not Lex-Ex	tendable?	
Extend	ability Check	[Ruppert & Seidel 1992]: Extendability of partial triang	ulations is Nf	P-complete.
Obser	vation	Each interior facet must be covered by additional simplices to complete a triangulation.		
Globa	l Data	Preprocess for each simplex s		
	•	its interior facets $\leadsto \mathcal{F}(s)$		
	•	all simplices with proper inter	section $\rightsquigarrow \mathcal{A}$	$\mathcal{L}(s)$

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

The Problem 000	Structures fo	or Counting Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		Is a Subset Not Lex-Ex	tendable?	
Extend	lability Check	[Ruppert & Seidel 1992]: Extendability of partial triang	gulations is NI	P-complete.
Observation Ea si		Each interior facet must be covered by additional simplices to complete a triangulation.		
Globa	al Data	Preprocess for each simplex s		
		its interior facets $\leadsto \mathcal{F}(s)$		
	•	all simplices with proper inte	rsection $\rightsquigarrow \mathcal{F}$	$\mathfrak{l}(s)$
Loca	al Data	With each partial triangulation	on T store in l	ex-order:

e Problem Structures fo	Counting Structures for Counting Subsets	New Results 0000●00000	Conclusions/Questions
	Is a Subset Not Lex-E	xtendable?	
Extendability Check	[Ruppert & Seidel 1992]: Extendability of partial trian	gulations is NI	P-complete.
Observation	Each interior facet must be c simplices to complete a trian	overed by add gulation.	itional
Global Data	Preprocess for each simplex a	S	
•	its interior facets $\leadsto \mathcal{F}(s)$		
•	all simplices with proper inte	ersection $\rightsquigarrow \mathcal{P}$	$\mathfrak{l}(s)$
Local Data	With each partial triangulati	on T store in I	ex-order:
Extendability Check Observation Global Data • • Local Data	[Ruppert & Seidel 1992]: Extendability of partial trians Each interior facet must be c simplices to complete a trian Preprocess for each simplex s its interior facets $\rightsquigarrow \mathcal{F}(s)$ all simplices with proper inte With each partial triangulati	gulations is NI overed by add gulation. s ersection $\rightsquigarrow \mathcal{P}$ on <i>T</i> store in I	P-complete litional I(s) ex-order:

The Problem 000	Structures fo	or Counting Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		Is a Subset Not Lex-Ex	tendable?	
Extend	ability Check	[Ruppert & Seidel 1992]: Extendability of partial triange	ulations is NF	P-complete.
Obser	vation	Each interior facet must be co simplices to complete a triang	vered by add ulation.	itional
Globa	l Data	Preprocess for each simplex s		
	•	its interior facets $\rightsquigarrow \mathcal{F}(s)$		
	•	all simplices with proper inter-	section $\rightsquigarrow \mathcal{A}$	l(s)
Loca	l Data	With each partial triangulatio	n T store in	ex-order:
	•	the free interior facets $\rightsquigarrow \mathcal{F}(T)$	Г)	

The Problem 000	Structures fo	or Counting	Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		ls a S	ubset Not Lex-E	Extendable?	
Extend	ability Check	[Ruppe Extenda	rt & Seidel 1992]: ability of partial tria	ngulations is N	P-complete.
Obser	vation	Each in simplice	terior facet must be es to complete a tria	covered by add ngulation.	litional
Globa	l Data	Preproc	cess for each simplex	: S	
	•	its inter	rior facets $\rightsquigarrow \mathcal{F}(s)$		
	►	all simp	olices with proper int	tersection $\rightsquigarrow \mathcal{F}$	$\mathfrak{A}(s)$
Loca	l Data	With ea	ach partial triangulat	tion T store in	lex-order:
	•	the free	e interior facets $\leadsto \mathcal{F}$	$\tilde{T}(T)$	
	•	the grea	ater simplices that ir	ntersect properl	$y \rightsquigarrow \mathcal{A}(T)$

The Problem Structures fo	r Counting Structures for Counting Subsets 000	New Results 000000000	Conclusions/Questions 000
	Is a Subset Not Lex-Ex	tendable?	
Extendability Check	[Ruppert & Seidel 1992]: Extendability of partial triang	ulations is NI	P-complete.
Observation	Each interior facet must be co simplices to complete a triang	overed by add gulation.	itional
Global Data	Preprocess for each simplex s		
•	its interior facets $\leadsto \mathcal{F}(s)$		
•	all simplices with proper inter	\cdot section $\rightsquigarrow \mathcal{F}$	$\mathfrak{l}(s)$
Local Data	With each partial triangulation	on T store in I	ex-order:
•	the free interior facets $\leadsto \mathcal{F}($	<i>T</i>)	
•	the greater simplices that inte	ersect properl	$y \rightsquigarrow \mathcal{A}(T)$
Observation	A partial triangulation \overline{a}	Г is <mark>not</mark> lex-ex =	tendable
	there is $\{f \in \mathcal{F}(T)\}$ not con	tained in any	$\{s \in \mathcal{A}(T)\}.$

	Structures for Counting Subsets	New Results	Conclusions/Questions
		0000000000	

Ingredient II: Lex-pruning/Lex-Breaking

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

◆□▶ ◆□▶ ◆□▶ ◆□▶ □ のへで

Ingredient II: Lex-pruning/Lex-Breaking

Theorem

A partial triangulation T with free interior facets $\mathcal{F}(T)$ and properly intersecting greater simplices $\mathcal{A}(T)$ is not lex-extendable to a triangulation

 $\min\{f \in \mathcal{F}(T)\} < \min\{\mathcal{F}(\min\{s \in \mathcal{A}(T)\})\}.$

◆□▶ ◆□▶ ◆□▶ ◆□▶ □ のへで

Ingredient II: Lex-pruning/Lex-Breaking

Theorem

A partial triangulation T with free interior facets $\mathcal{F}(T)$ and properly intersecting greater simplices $\mathcal{A}(T)$ is not lex-extendable to a triangulation

 $\min\{f \in \mathcal{F}(T)\} < \min\{\mathcal{F}(\min\{s \in \mathcal{A}(T)\})\}.$

Theorem

A partial triangulation T with free interior facets $\mathcal{F}(T)$ and properly intersecting greater simplices $\mathcal{A}(T)$ is not lex-extendable to a triangulation starting with some $s' \ge s$ in $\mathcal{A}(T)$

 $\min\{f \in \mathcal{F}(T)\} < \min\{\mathcal{F}(s)\}.$

Ingredient II: Lex-pruning/Lex-Breaking

Theorem

A partial triangulation T with free interior facets $\mathcal{F}(T)$ and properly intersecting greater simplices $\mathcal{A}(T)$ is not lex-extendable to a triangulation

 $\min\{f \in \mathcal{F}(T)\} < \min\{\mathcal{F}(\min\{s \in \mathcal{A}(T)\})\}.$

Theorem

A partial triangulation T with free interior facets $\mathcal{F}(T)$ and properly intersecting greater simplices $\mathcal{A}(T)$ is not lex-extendable to a triangulation starting with some $s' \ge s$ in $\mathcal{A}(T)$

 $\min\{f \in \mathcal{F}(T)\} < \min\{\mathcal{F}(s)\}.$

◆□▶ ◆□▶ ◆□▶ ◆□▶ □ のへで

Gain One integer comparison instead of many subset tests.

or Counting Structures for	r Counting subsets New Results	Conclusions/Questions
	0000000000000000000000000000000000	

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ● □ ● ● ● ●

Effectivity



Effectivity

Comparison (with lex-breaking):



◆□▶ ◆□▶ ◆ □▶ ◆ □▶ - □ - のへぐ



ヘロト 人間 とくほとくほとう

э.





Computational Results for Triangulations

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ つへぐ

	Structures for Counting Subsets	New Results	Conclusions/Questions
		0000000000	

Computational Results for Triangulations

MPTOPCOM Flip-Based CPU Times (16/40 Threads)

[Jordan & Joswig & Kastner 2018]

Point Conf.	# Triang's	# Orbits	CPU time [hh:mm:ss]
$[0, 1]^4$ $3D_3$ (reg./full/output)	92,487,256 22,201,684,367	247,451 925,148,763	00:01:56 96:00:00

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

MPTOPCOM Flip-Based **CPU** Times (16/40 Threads)

_

TOPCOM 1.0.8 Subset-Based **CPU** Times (16 Threads)

Computational Results for Triangulations

[Jordan & Joswig & Kastner 2018]

Point Conf.	# Triang's # Orbits		CPU time [hh:mm:ss]
[0, 1] ⁴	92,487,256	247,451	00:01:56
3D ₃	22,201,684,367	925,148,763	96:00:00
(reg./full/output)			
[R. 2022]			
Point Conf.	# Triang's	# Orbits	CPU time [hh:mm:ss]
[0, 1] ⁴	92,487,256	247,451	00:00:04
$3D_3$ (output)	22,201,684,367	925,148,763	01:05:11
$3D_3$ (count)	22,201,684,367	925,148,763	00:21:02
3D ₃ (regular)	21,861,522,799	910,974,879	20:21:53
$3D_3$ (full)	511,052,427	21,302,400	00:01:01
$3D_3$ (unimod.)	346,903,379	14,459,488	00:00:39
Dodecahedron	1,533,079,037,570	12,775,757,027	11:11:48
Pyritohedron	32,734,029,351,118	1,363,918,758,719	692:30:04
$\Delta_5 \times \Delta_3$	442,472,050,753,920	25,606,173,722	1313:57:17
			(M1Max8t)

The Problem 000	Structures f 0000		Structures for Counting Subsets	New Results 0000000000	Conclusions/Questions
		Bonu	is Track I		

▲□▶ ▲圖▶ ▲ 臣▶ ▲ 臣▶ 臣 - のへで








For Raman

Triangulations with only simplices of min. vol. of generalized hypersimplices [Manecke et al. 2020]:

- $\Delta(5, 1, 3)$ has 27,780 (250 classes)
- $\Delta(5, 1, 4)$ has 5 (1 class)
- $\Delta(6, 1, 3)$ has more than 245,074,320 (340,381 classes)

Bonus Track I

For Raman

Triangulations with only simplices of min. vol. of generalized hypersimplices [Manecke et al. 2020]:

- $\Delta(5, 1, 3)$ has 27,780 (250 classes)
- ► Δ(5, 1, 4) has 5 (1 class)
- Δ(6, 1, 3) has more than 245,074,320 (340,381 classes)
- Δ(6, 1, 4) has more than 249,295,320 (347,613 classes)

Bonus Track I

For Raman

Triangulations with only simplices of min. vol. of generalized hypersimplices [Manecke et al. 2020]:

- $\Delta(5, 1, 3)$ has 27,780 (250 classes)
- ► Δ(5, 1, 4) has 5 (1 class)
- Δ(6, 1, 3) has more than 245,074,320 (340,381 classes)
- Δ(6, 1, 4) has more than 249,295,320 (347,613 classes)
- ▲(6, 2, 4) has more than 7,248,961,080 (10,068,279 classes)

The Problem 000	Structures for Counting		Structures for Counting Subsets	New Results 000000000	Conclusions/Questions
		Bonus	Track II		

▲□▶▲圖▶▲≣▶▲≣▶ ■ 少々⊙

Other Results Enumeration of (co-)circuits (different lex-min check):

◆□▶ ◆□▶ ◆臣▶ ◆臣▶ ─臣 ─のへで

Other Results

Enumeration of (co-)circuits (different lex-min check): [0, 1]⁸ has 38,636,185,528,212,416 circuits in 3,858,105,362 classes (CPU: 163:37:00) (asked by Lisa Lamberti and Komei Fukuda)

Other Results

Enumeration of (co-)circuits (different lex-min check):

 [0, 1]⁸ has 38,636,185,528,212,416 circuits in 3,858,105,362 classes (CPU: 163:37:00) (asked by Lisa Lamberti and Komei Fukuda)
 [0,1]⁹ has 448,691,419,804,586 cocircuits in 3,899,720 classes (CPU: 13:30:12)

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

(extends [Aichholzer & Aurnhammer 1996])

Other Results

Enumeration of (co-)circuits (different lex-min check):

 [0, 1]⁸ has
 38,636,185,528,212,416 circuits in 3,858,105,362 classes (CPU: 163:37:00)
 (asked by Lisa Lamberti and Komei Fukuda)
 [0,1]⁹ has
 448,691,419,804,586 cocircuits in 3,899,720 classes (CPU: 13:30:12)
 (extends [Aichholzer & Aurnhammer 1996])

▲ロト ▲ □ ト ▲ □ ト ▲ □ ト ● ● の Q ()

Sideline Necessary conditions for lex-extendability

Other Results

Enumeration of (co-)circuits (different lex-min check): $[0, 1]^8$ has 38,636,185,528,212,416 circuits in 3,858,105,362 classes (CPU: 163:37:00) (asked by Lisa Lamberti and Komei Fukuda) [0,1]⁹ has 448,691,419,804,586 cocircuits in 3,899,720 classes

(CPU: 13:30:12) (extends [Aichholzer & Aurnhammer 1996])

Sideline Necessary conditions for lex-extendability

found for cocircuits

Other Results

 Enumeration of (co-)circuits (different lex-min check):
 [0, 1]⁸ has 38,636,185,528,212,416 circuits in 3,858,105,362 classes (CPU: 163:37:00) (asked by Lisa Lamberti and Komei Fukuda)

 [0,1]⁹ has 448,691,419,804,586 cocircuits in 3,899,720 classes (CPU: 13:30:12) (extends [Aichholzer & Aurnhammer 1996])

Sideline Necessary conditions for lex-extendability

- found for cocircuits
- but not so far for circuits.



				Conclusions/Questions ○●○
	Conc	lusions/Question	S	

▲□▶▲圖▶▲≣▶▲≣▶ ≣ のへで













Structures for Counting Subsets 000

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>

Conclusions/Questions

Conclusions

- Enumeration of orbits of triangulations accelerated by "Geometry meets Combinatorics":
- critical-element tables for lex-min check
- minimal-element comparison for lex-extendability check

Questions Potential further research:

- Investigate the complexity of symLSRS.
- Apply symLSRS to more examples.
- Represent flip-graph exploration in terms of subsets.