3. Aggregate Quantifier Complexity

Camilo Thorne[1]

[1]KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano
cthorne@inf.unibz.it
http://www.inf.unibz.it/~cathorne

ESSLLI 2013, Aug 5-9, Düsseldorf

Outline

http://www.inf.unibz.it/~cathorne/agg_esslli

Motivation

$\models$

more than one third of people are women

the average height of women is 165cm

every person is a man

some person is a woman

Motivation

$\models$

more than one third of people are women

the average height of women is 165cm

every person is a man

some person is a woman

Q: How complex?

Semantic Complexity [PH10]

Definition (Semantic Complexity)

Given model $\mathcal{I}$, the semantic complexity of quantified sentence S is defined as the cost of computing $\mathcal{I}, \gamma \models \tau(S)$, for some $\gamma \in \Delta^{FV(\tau(S))}$

- We have two cases:
  1. $\tau(S)$ in FOL: $\mathcal{I}, \gamma \models \phi$
  2. $\tau(S)$ in A-FOL: $\mathcal{I}, \gamma \models \varphi$
- A-FOL more expressive than FOL, we expect A-FOL to be more complex
- Fine-grained analysis possible:
  1. if measured only in $\#(\Delta)$: data complexity
  2. otherwise: combined complexity

Turing Machines [Pap94]

Definition (Turing Machine)

A Turing machine is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_s, q_f)$ where:

1. $\Sigma = \{0, 1\}$ is the input alphabet
2. $Q = \{q_s, q_0, \ldots, q_n, q_f\}$ the set of (internal) states
3. q_s and q_f the initial and accepting state, resp.
4. Γ the tape alphabet
5. $\delta \subseteq (Q \times (\Gamma \cup \Sigma)) \times (Q \times (\Gamma \cup \Sigma) \times \{R, L, H\})$ the transition relation

Turing Machines [Pap94]

Definition (Turing Machine)

A Turing machine is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_s, q_f)$ where:

1. $\Sigma = \{0, 1\}$ is the input alphabet
2. $Q = \{q_s, q_0, \ldots, q_n, q_f\}$ the set of (internal) states
3. q_s and q_f the initial and accepting state, resp.
4. Γ the tape alphabet
5. $\delta \subseteq (Q \times (\Gamma \cup \Sigma)) \times (Q \times (\Gamma \cup \Sigma) \times \{R, L, H\})$ the transition relation

- Two kinds:

Turing Machines [Pap94]

Definition (Turing Machine)

A Turing machine is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_s, q_f)$ where:

1. $\Sigma = \{0, 1\}$ is the input alphabet
2. $Q = \{q_s, q_0, \ldots, q_n, q_f\}$ the set of (internal) states
3. q_s and q_f the initial and accepting state, resp.
4. Γ the tape alphabet
5. $\delta \subseteq (Q \times (\Gamma \cup \Sigma)) \times (Q \times (\Gamma \cup \Sigma) \times \{R, L, H\})$ the transition relation

- Two kinds:
  1. if δ functional: deterministic Turing machine
  2. otherwise: non-deterministic Turing machine

Turing Machines [Pap94]

Definition (Turing Machine)

A Turing machine is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_s, q_f)$ where:

1. $\Sigma = \{0, 1\}$ is the input alphabet
2. $Q = \{q_s, q_0, \ldots, q_n, q_f\}$ the set of (internal) states
3. q_s and q_f the initial and accepting state, resp.
4. Γ the tape alphabet
5. $\delta \subseteq (Q \times (\Gamma \cup \Sigma)) \times (Q \times (\Gamma \cup \Sigma) \times \{R, L, H\})$ the transition relation

- Two kinds:
  1. if δ functional: deterministic Turing machine
  2. otherwise: non-deterministic Turing machine

- We can model the time and space required by computations:

Turing Machines [Pap94]

Definition (Turing Machine)

A Turing machine is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_s, q_f)$ where:

1. $\Sigma = \{0, 1\}$ is the input alphabet
2. $Q = \{q_s, q_0, \ldots, q_n, q_f\}$ the set of (internal) states
3. q_s and q_f the initial and accepting state, resp.
4. Γ the tape alphabet
5. $\delta \subseteq (Q \times (\Gamma \cup \Sigma)) \times (Q \times (\Gamma \cup \Sigma) \times \{R, L, H\})$ the transition relation

- Two kinds:
  1. if δ functional: deterministic Turing machine
  2. otherwise: non-deterministic Turing machine

- We can model the time and space required by computations:
  1. computation steps $\Rightarrow$ time $t(M)$ of M
  2. size of tape $\Rightarrow$ space $s(M)$ of M

Example of Turing Machine

- The following Turing machine M_2 decides the problem

$$\mathcal{P}_2 = \{2^i \mid i \geq 1\} = \{2^1, 2^2, 2^3, \dots\}$$

(in binary: 1 followed by 0 i-times, for $i \geq 1$)

- $M_2 = (\{1, 0\}, \{*\}, Q_2, \delta_2, q_s, q_f)$ where:

$$Q_2 = \{q_s, q_1, q_f\}$$

δ_2 defined by :

$$\begin{array}{ll} \delta_2(q_s, 0) = (q_s, 0, H) & \delta_2(q_s, 1) = (q_1, 1, R) \\ \delta_2(q_1, 0) = (q_1, 0, R) & \delta_2(q_1, 1) = (q_1, 1, H) \\ \multicolumn{2}{c}{\delta_2(q_1, *) = (q_f, 1, H)} \end{array}$$

Decision Problems and Classes [Pap94]

Definition (Decidability)

(1) M decides $\mathcal{P}$ iff for all $w \in \{0,1\}^*$, M returns "yes" on w whenever $w \in \mathcal{P}$
(2) If such M exists for $\mathcal{P}$, $\mathcal{P}$ is decidable

- The time $t(M)$ and space $s(M)$ that M spends on deciding $\mathcal{P}$ give way to the time and space complexity of $\mathcal{P}$
- Problems can be classified into complexity classes C depending on (1) the function describing $t(M)$ and $s(M)$ and (2) the type of M
  1. M det., $t(M) \sim O(n^k)$: P
  2. M non-det., $t(M) \sim O(n^k)$: NP
  3. M det,. $s(M) \sim O(\log n)$: L
  4. M non-det,. $s(M) \sim O(\log n)$: NL
  5. ...
- They form a hierarchy: $\mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \ldots$

C-hard and C-complete Problems [Pap94]

Definition (C-completeness)

A problem $\mathcal{P}$ is said to be:

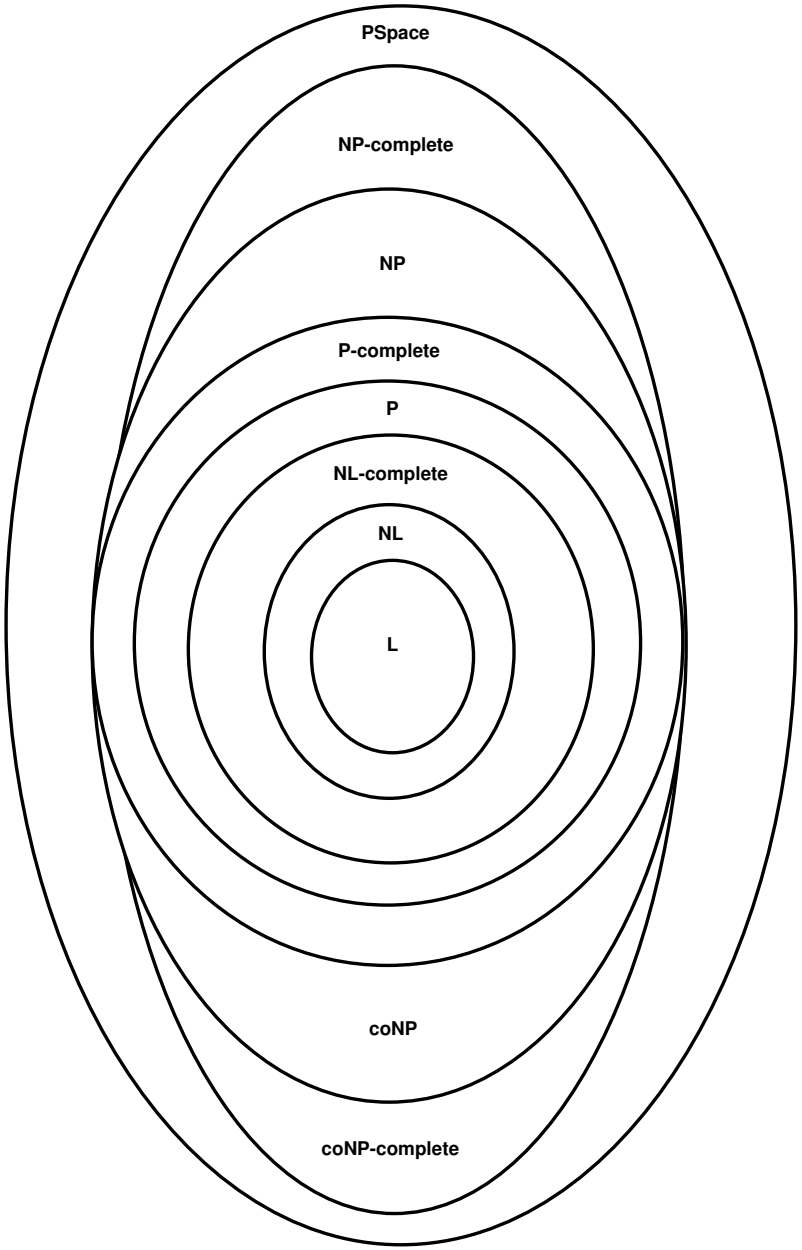
1. C-hard if it is as hard as every problem $\mathcal{P}'$ in C
2. C-complete if it is C-hard and also in C

- If $\mathrm{C}' \subseteq \mathrm{C}$ and $\mathcal{P}$ is C-hard/complete, then $\mathcal{P} \in \mathrm{C} \setminus \mathrm{C}'$
- C-hard/complete problems are "intrinsecally" in C

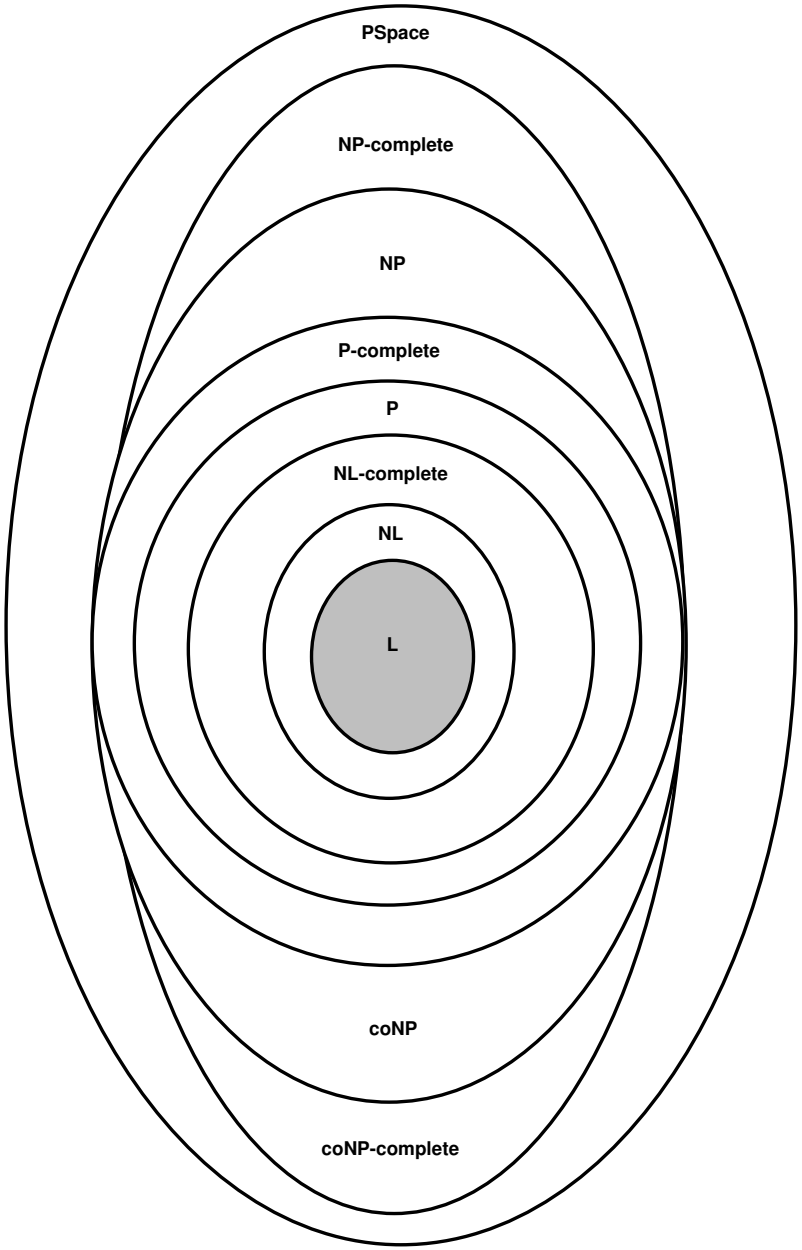
Remark

By "$\mathcal{P}$ as hard as $\mathcal{P}'$" we understand that there exists a Turing machine M that reduces $\mathcal{P}$ to $\mathcal{P}'$ using logarithmic space!

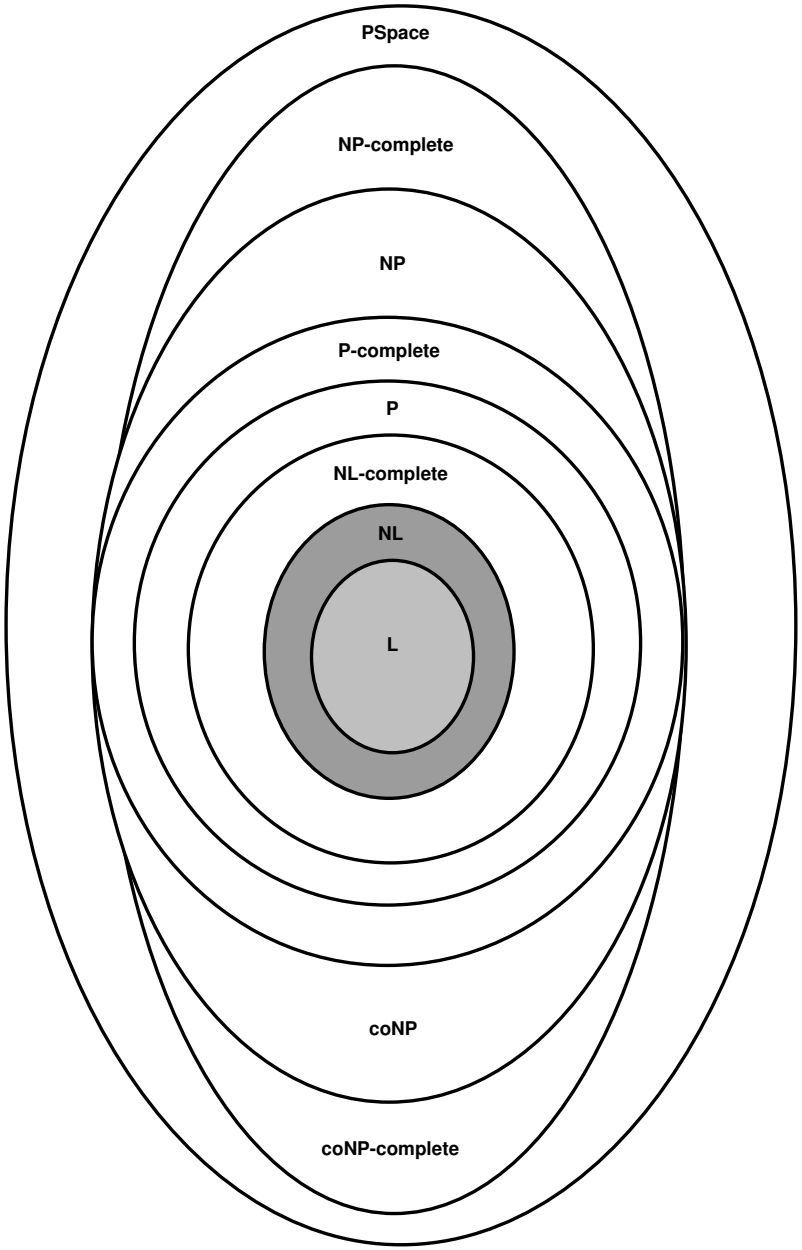
Tractability and Intractability [Pap94]



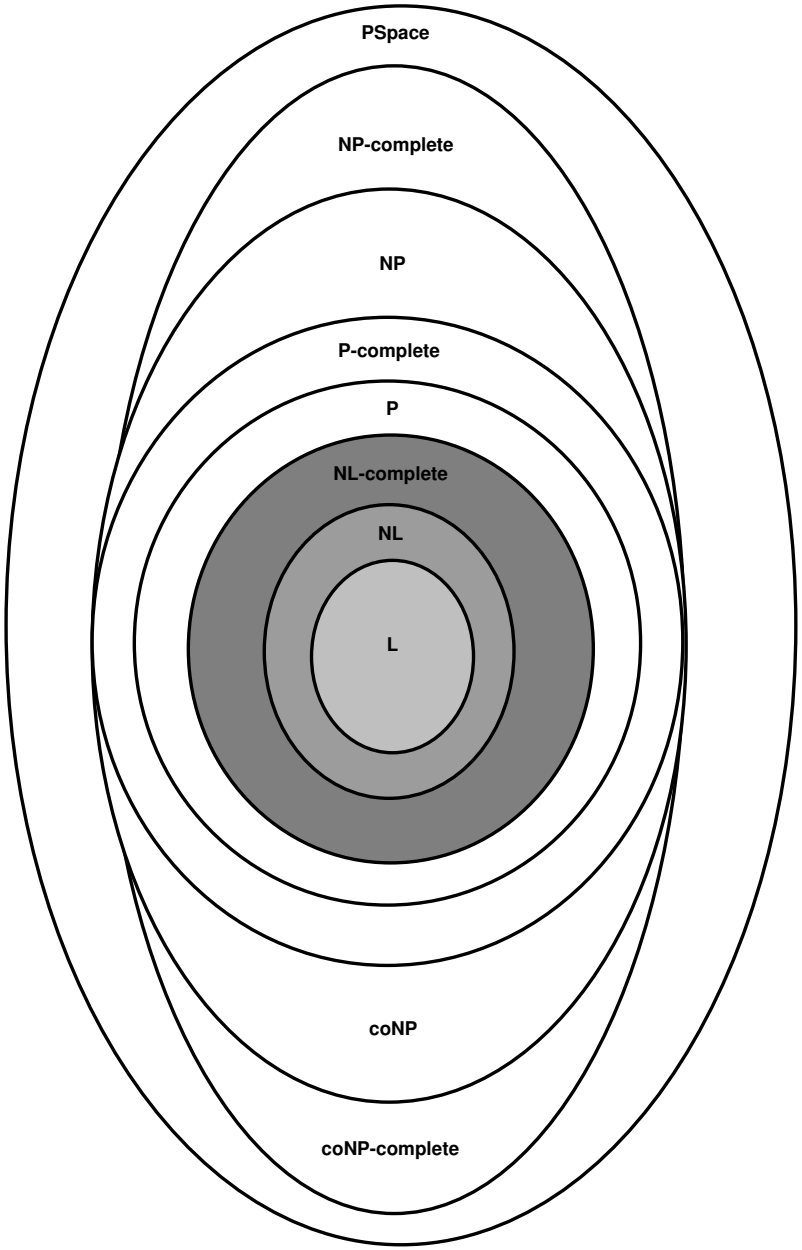
Tractability and Intractability [Pap94]



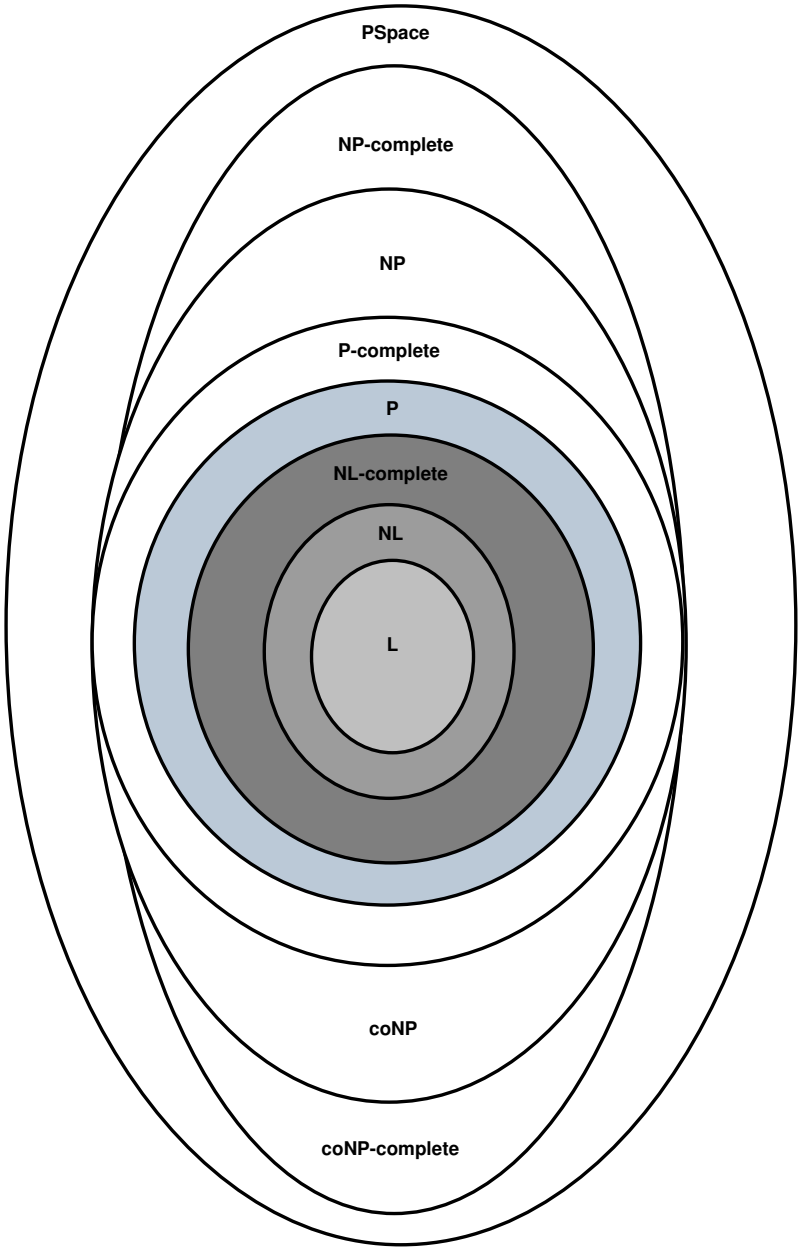
Tractability and Intractability [Pap94]



Tractability and Intractability [Pap94]

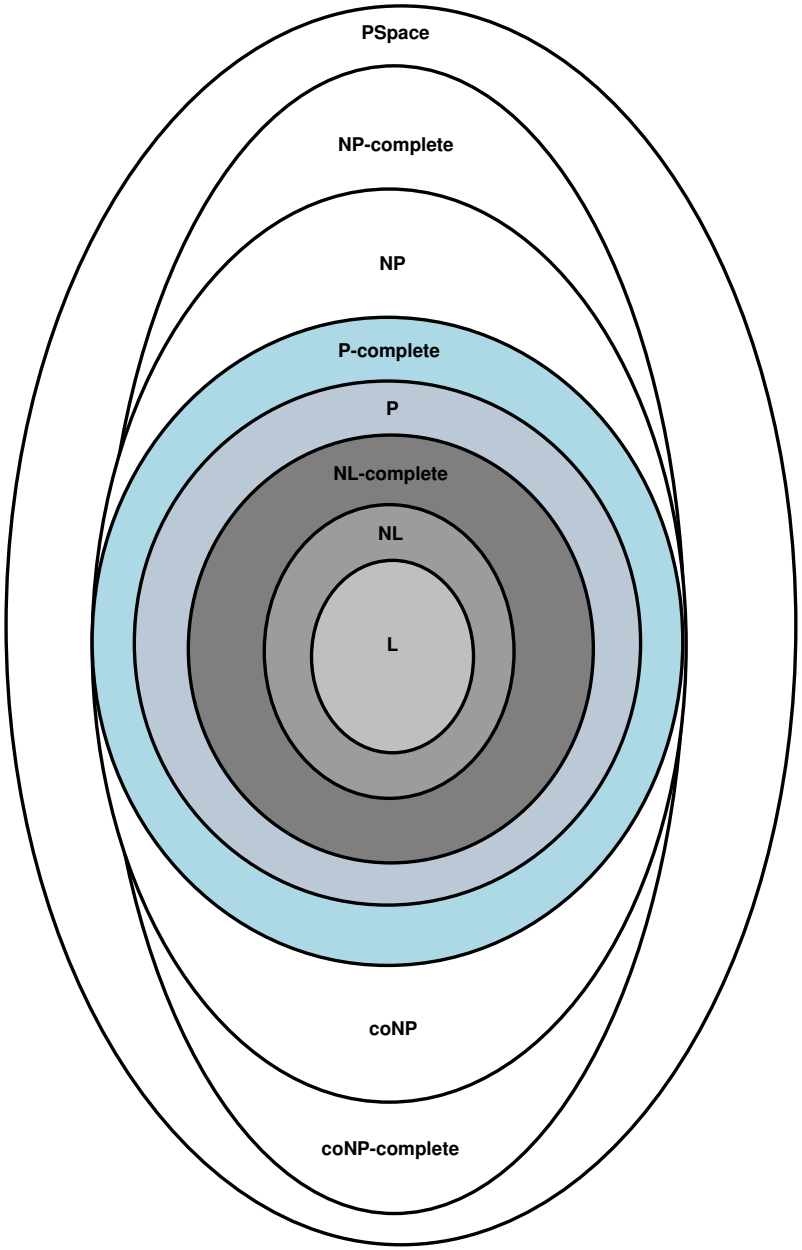


Tractability and Intractability [Pap94]



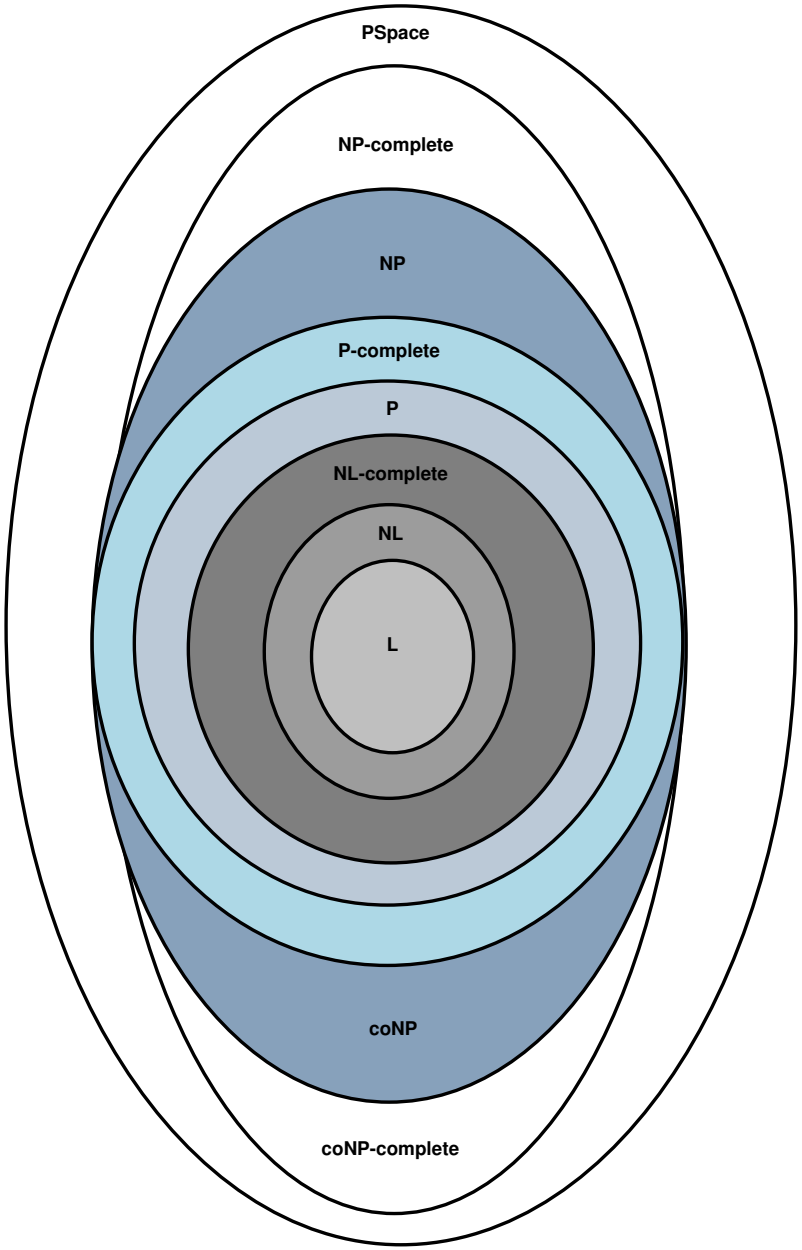
- Problems in P are tractable

Tractability and Intractability [Pap94]



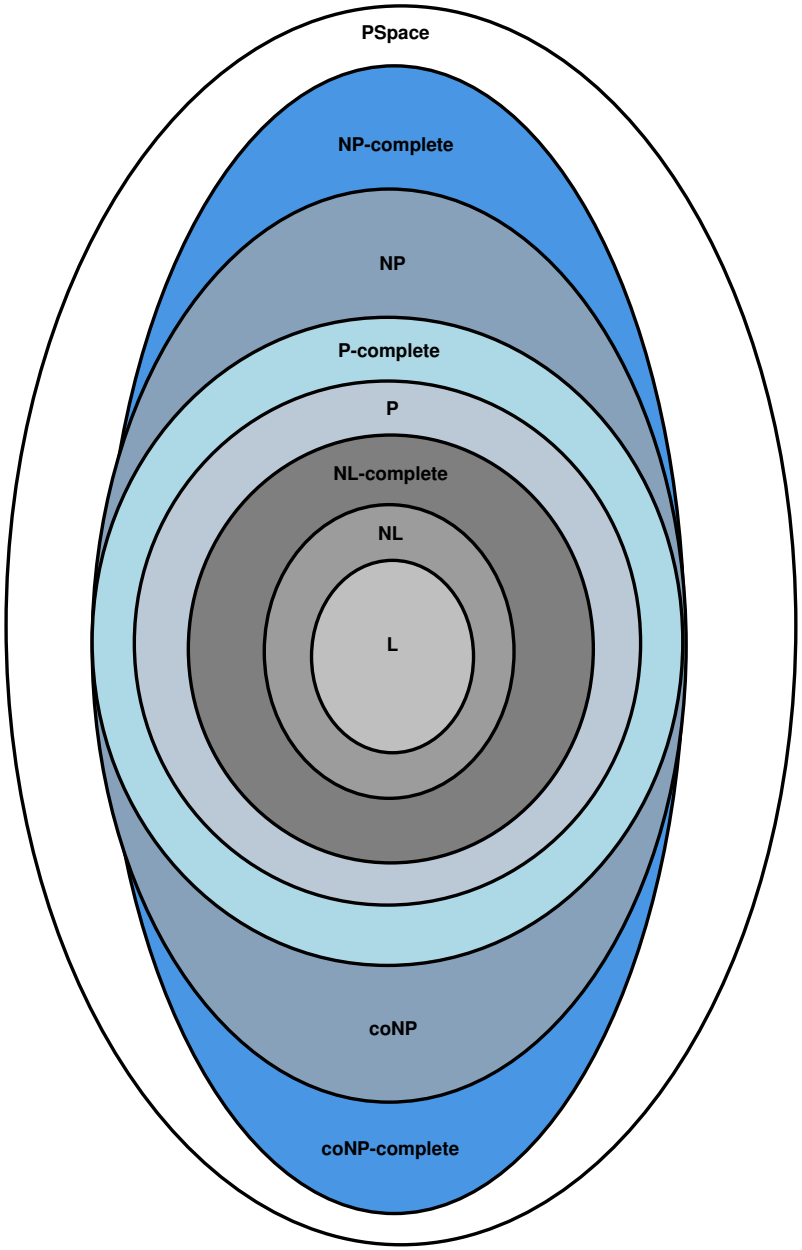
- Problems in P are tractable

Tractability and Intractability [Pap94]



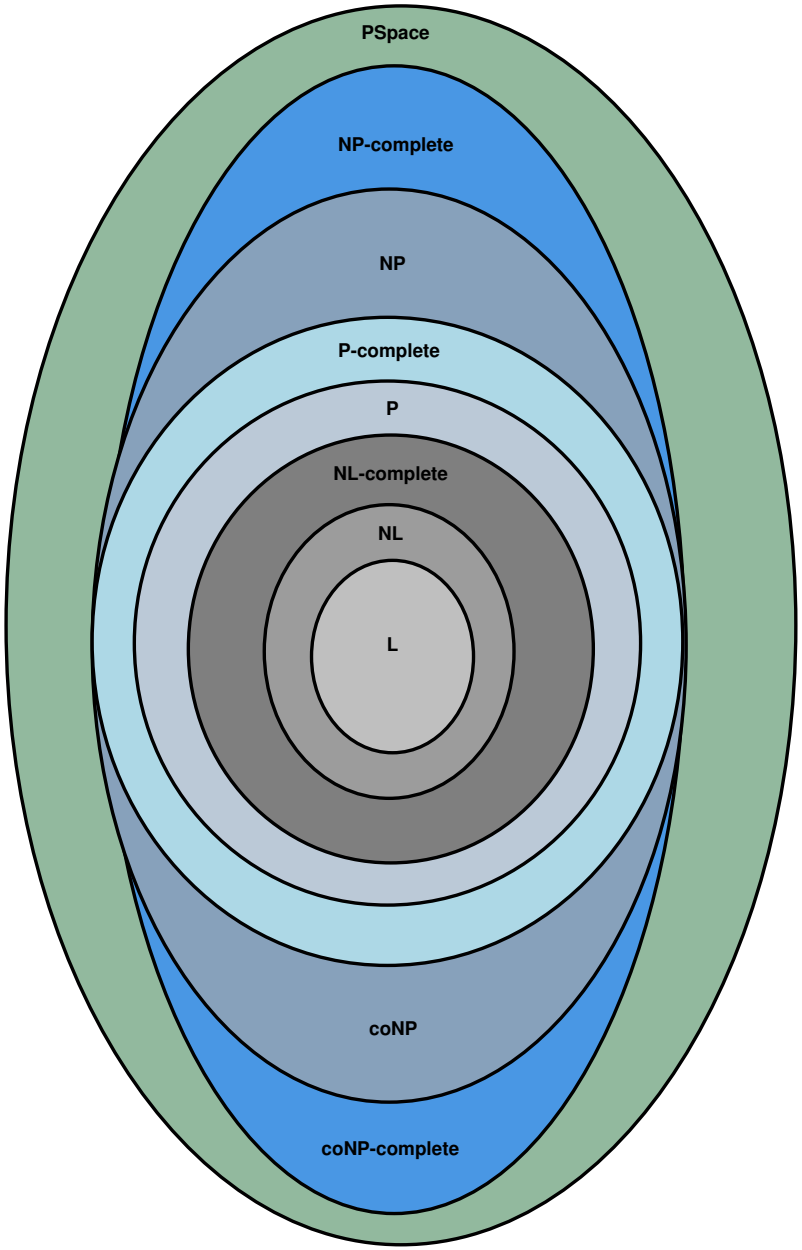
- Problems in P are tractable

Tractability and Intractability [Pap94]



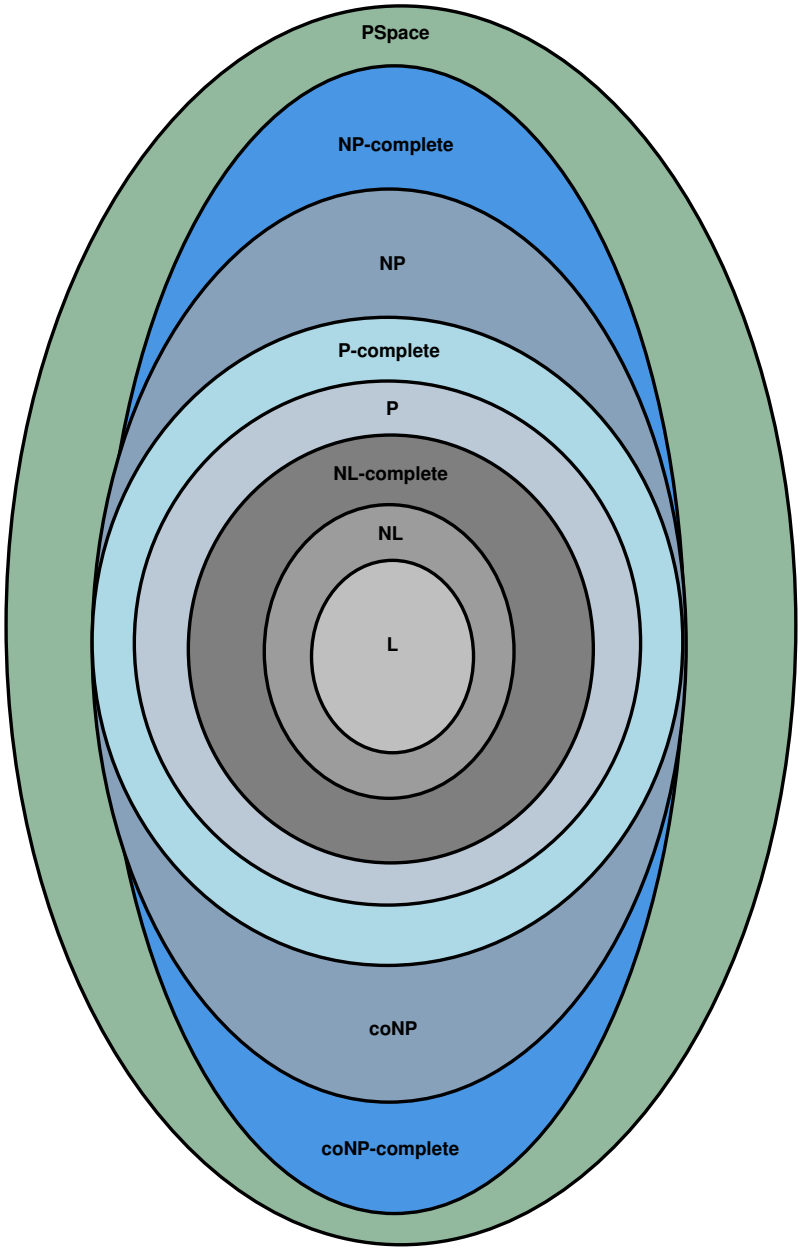
- Problems in P are tractable
- NP-hard problems are intractable ("intrinsecally" exponential)

Tractability and Intractability [Pap94]



- Problems in P are tractable
- NP-hard problems are intractable ("intrinsecally" exponential)
- Beyond PSPACE, problems become harder and eventually, unsolvable

Tractability and Intractability [Pap94]



- Problems in P are tractable
- NP-hard problems are intractable ("intrinsecally" exponential)
- Beyond PSPACE, problems become harder and eventually, unsolvable

Remarks

(1) We conjecture that $\mathrm{P} \neq \mathrm{NP}$
(2) $\mathrm{coC} = \{\{0,1\}^* \setminus \mathcal{P} \mid \mathcal{P} \in \mathrm{C}\}$

Computing Aggregations in $O(\log \#(\Delta))$ Space

1: **procedure** $\textsc{Ans}_\alpha(Q(\alpha(\beta(P))), \mathcal{I})$
2: $\quad \varphi(x)_P \leftarrow \textsc{Core}(Q(\alpha(\beta(P))))$; ▷ compute core
3: $\quad s \leftarrow 0$; $a \leftarrow 0$; $n \leftarrow 0$; $p \leftarrow 0$; ▷ initialize
4: $\quad$ **for** $\gamma \in \mathit{Sat}_\mathcal{I}(\varphi(x))$ **do** ▷ $\mathit{Sat}_\mathcal{I}(\varphi(x)) = \{\gamma \mid \mathcal{I}, \gamma \models \varphi(x)\}$
5: $\quad\quad n \leftarrow n + 1$; $s \leftarrow s + \beta(\gamma(x))$; ▷ update
6: $\quad\quad a \leftarrow \frac{s}{n}$; $p \leftarrow p \times \beta(\gamma(x))$; ▷ update
7: $\quad\quad$ **if** $\alpha = \mathbf{count}$ and $Q(n)$ **then** ▷ test
8: $\quad\quad\quad$ **return** true;
9: $\quad\quad$ **else**
10: $\quad\quad\quad$ **if** $\alpha = \mathbf{avg}$ and $Q(a)$ **then** ▷ test
11: $\quad\quad\quad\quad$ **return** true;
12: $\quad\quad\quad$ **else**
13: $\quad\quad\quad\quad$ **if** $\alpha = \mathbf{sum}$ and $Q(s)$ **then** ▷ test
14: $\quad\quad\quad\quad\quad$ **return** true;
15: $\quad\quad\quad\quad$ **else**
16: $\quad\quad\quad\quad\quad$ **if** $\alpha = \mathbf{prod}$ and $Q(p)$ **then** ▷ test
17: $\quad\quad\quad\quad\quad\quad$ **return** true;
18: $\quad\quad\quad\quad\quad$ **end if**
19: $\quad\quad\quad\quad$ **end if**
20: $\quad\quad\quad$ **end if**
21: $\quad\quad$ **end if**
22: $\quad$ **end for**
23: $\quad$ **return** false; ▷ false if all tests fail
24: **end procedure**

Semantic Complexity of Aggregation

Theorem

The semantic (data) complexity of FOL quantifiers is in AC^0

Lemma

The algorithm $\textsc{Ans}_\alpha(\cdot,\cdot)$ *is sound and complete, that is, for all interpretations* $\mathcal{I}$*, each A-FOL formula* ϕ *and* $\gamma(\cdot) : FV(\phi) \to \Delta$

$$\mathcal{I}, \gamma \models \phi \quad \textit{iff} \quad \textsc{Ans}_\alpha(\mathcal{I}, \phi) = true$$

Theorem

The semantic (data) complexity of aggregate quantifiers (and proportional quantifiers) is in L

Quantifier Complexity [TS13]

Quantifier	Semantics	D.C.
some	$\{(A,B) \subseteq \Delta \times \Delta \mid A \cap B \neq \emptyset\}$	AC^0
every	$\{(A,B) \subseteq \Delta \times \Delta \mid A \subseteq B\}$	AC^0
at least k	$\{(A,B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \geq k\}$	AC^0
more than k	$\{(A,B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \geq k+1\}$	AC^0
exactly k	$\{(A,B) \subseteq \Delta \times \Delta \mid \#(A \cap B) = k\}$	AC^0
the total α of	$\{(A,B) \subseteq \Delta \times \Delta \mid \mathbf{sum}(\mu_\alpha(A)) \in B\}$	L
the number of	$\{(A,B) \subseteq \Delta \times \Delta \mid \mathbf{count}(A) \in B\}$	L
the α-est	$\{(A,B) \subseteq \Delta \times \Delta \mid \mathbf{argmax}(\mu_\alpha(A)) \in B\}$	AC^0
the average α of	$\{(A,B) \subseteq \Delta \times \Delta \mid \mathbf{avg}(\mu_\alpha(A)) \in B\}$	L
the product α of	$\{(A,B) \subseteq \Delta \times \Delta \mid \mathbf{prod}(\mu_\alpha(A)) \in B\}$	L
most	$\{(A,B) \subseteq \Delta \times \Delta \mid \#(A \cap B) > \#(A \setminus B)\}$	L
more than p/k of	$\{(A,B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \geq p \cdot (\#(A)/k)\}$	L

Ramsey Quantifiers [Szy09, BSS11]

Definition (Ramseyfication)

Let Q be a quantifier of type $(1,1)$. The Ramseyfication of Q is $R_Q = \{(A, R) \subseteq \Delta \times (\Delta \times \Delta) \mid \text{exists } X \subseteq A \text{ s.t. } (A, X) \in Q, \text{ and for all } x, y \in X, (x, y) \in R\}$ of type $(1,2)$.

- Turns quantifiers of type $(1,1)$ into a polyadic quantifiers of type $(1,2)$,
- Conveyed in English by the reciprocal **NP** "each other" under default (strong) interpretation.
- It intuitively states that $\mathcal{I}$ is a graph with connected components.

Ramsey Quantifiers [Szy09, BSS11]

Definition (Ramseyfication)

Let Q be a quantifier of type $(1,1)$. The Ramseyfication of Q is $R_Q = \{(A,R) \subseteq \Delta \times (\Delta \times \Delta) \mid \text{exists } X \subseteq A \text{ s.t. } (A,X) \in Q, \text{ and for all } x,y \in X, (x,y) \in R\}$ of type $(1,2)$.

- Turns quantifiers of type $(1,1)$ into a polyadic quantifiers of type $(1,2)$,
- Conveyed in English by the reciprocal **NP** "each other" under default (strong) interpretation.
- It intuitively states that $\mathcal{I}$ is a graph with connected components.

Theorem ([Szy10])

If $\mathbf{Q}$ *is "at least* k*" or "more than* p/k*", then* R_Q *is* NP*-complete.*

Ramsey Quantifiers [Szy09, BSS11]

Definition (Ramseyfication)

Let Q be a quantifier of type $(1,1)$. The Ramseyfication of Q is $R_Q = \{(A, R) \subseteq \Delta \times (\Delta \times \Delta) \mid \text{exists } X \subseteq A \text{ s.t. } (A, X) \in Q, \text{ and for all } x, y \in X, (x, y) \in R\}$ of type $(1,2)$.

- Turns quantifiers of type $(1,1)$ into a polyadic quantifiers of type $(1,2)$,
- Conveyed in English by the reciprocal **NP** "each other" under default (strong) interpretation.
- It intuitively states that $\mathcal{I}$ is a graph with connected components.

Theorem ([Szy10])

If $\mathbf{Q}$ *is "at least* k*" or "more than* p/k*", then* R_Q *is* NP*-complete.*

Q: Any idea why?

Ramseyfied Quantifier Complexity [Szy10]

Quantifier	**D.C.**
some + each other	P
every + each other	P
at least k + each other	NP-complete
more than k + each other	NP-complete
exactly k + each other	P
most + each other	P
more than p/k of + each other	NP-complete

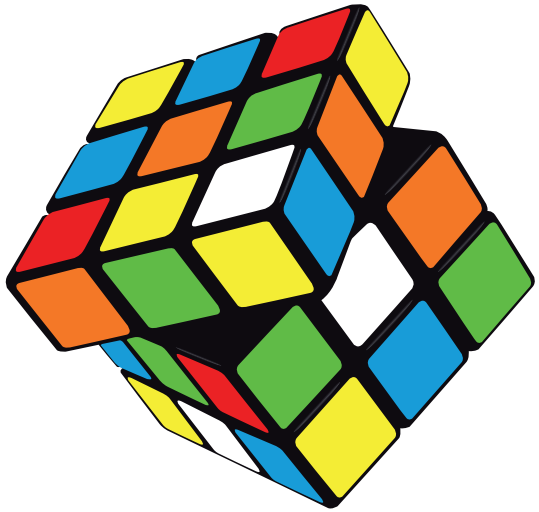
Ramseyfied Quantifier Complexity [Szy10]

Quantifier	D.C.
some + each other	P
every + each other	P
at least k + each other	NP-complete
more than k + each other	NP-complete
exactly k + each other	P
most + each other	P
more than p/k of + each other	NP-complete

Q: What about aggregations?

Summary

1. We have given an overview of the semantic complexity of generalized quantifiers
2. Semantic complexity was modeled as the complexity of model evaluation for A-FOL
3. We have shown that aggregate quantifiers are, in general, tractable
4. We have see that this is due to their being built over a FOL core
5. We have also recalled the basic notions of computability and complexity theory



Thank you :-)

References I

[BSS11] Oliver Bott, Fabian Schlotterbeck, and Jakub Szymanik.
Interpreting tractable versus intractable reciprocal sentences.
In *Proceedings of the 3rd Intenational Conference in Computational Semantics (IWCS 2011)*, 2011.

[Pap94] Christos Papadimitrou.
Computational Complexity.
Addison Wesley - Longman, 1994.

[PH10] Ian Pratt-Hartmann.
Computational complexity in natural language.
In *Handbook of Computational Linguistics and Natural Language Processing*, chapter 2, pages 43–73. Wiley-Blackwell, 2010.

[Szy09] Jakub Szymanik.
Quantifiers in Time and Space.
Institute for Logic, Language and Computation, 2009.

References II

[Szy10] Jakub Szymanik.
Computational complexity of polyadic lifts of generalized quantifiers in natural language.
Linguistics and Philosophy, 33(3):215–250, 2010.

[TS13] Camilo Thorne and Jakub Szymanik.
Quantifier distribution and semantic complexity.
In *Proceedings of the 2013 Tbilisi Colloquium (TbiLLC 2013)*, 2013.