TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science ● Institute of Theoretical Computer Science ● Chair of Automata Theory

NAVIGATING THE $\mathcal{EL}$ SUBSUMPTION HIERARCHY

Francesco Kriegel

Funded in DFG Project 430150274.

The 34th International Workshop on Description Logics (DL 2021), 20 September 2021

The $\mathcal{EL}$ Subsumption Hierarchy

- **Definition:** The $\mathcal{EL}$ *subsumption hierarchy* is the set of all $\mathcal{EL}$ concept descriptions, partially ordered by subsumption $\sqsubseteq_\emptyset$.
- One can navigate in this hierarchy by going up to subsumers and by going down to subsumees.

The $\mathcal{EL}$ Subsumption Hierarchy

- **Definition:** The $\mathcal{EL}$ *subsumption hierarchy* is the set of all $\mathcal{EL}$ concept descriptions, partially ordered by subsumption $\sqsubseteq_\emptyset$.
- One can navigate in this hierarchy by going up to subsumers and by going down to subsumees.
- How can smallest steps be made?
- **Definition:** C is a *lower neighbor* of D and D is an *upper neighbor* of C if
  - $C \sqsubset_\emptyset D$,
  - and there is no concept E such that $C \sqsubset_\emptyset E \sqsubset_\emptyset D$.

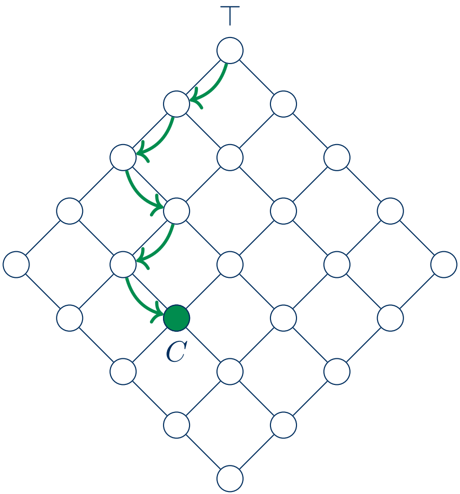
The $\mathcal{EL}$ Subsumption Hierarchy

- **Definition:** The $\mathcal{EL}$ *subsumption hierarchy* is the set of all $\mathcal{EL}$ concept descriptions, partially ordered by subsumption $\sqsubseteq_\emptyset$.
- One can navigate in this hierarchy by going up to subsumers and by going down to subsumees.
- How can smallest steps be made?
- **Definition:** C is a *lower neighbor* of D and D is an *upper neighbor* of C if
  - $C \sqsubset_\emptyset D$,
  - and there is no concept E such that $C \sqsubset_\emptyset E \sqsubset_\emptyset D$.
- What is this good for?
- Sometimes it is desired to find a certain target concept but it is unclear how to compute it directly.
  - Example: Concept learning
  - Example: Ontology repair

The $\mathcal{EL}$ Subsumption Hierarchy

Example: Concept Learning

- **Goal:** Find a (maximally general) concept C that satisfies some conditions.
- One might start with the most general concept $\top$ and subsequently go to a lower neighbor until a suitable concept C has been found.
- We will see later why such an approach is not feasible in practice.



The $\mathcal{EL}$ Subsumption Hierarchy

Example: Ontology Repair

- In order to resolve inconsistency or to remove an unwanted consequence, the classical approach to repairing an ontology is deleting enough axioms.
- More fine-grained repairs can be obtained by *weakening axioms* instead of removing them completely.

The $\mathcal{EL}$ Subsumption Hierarchy

Example: Ontology Repair

- In order to resolve inconsistency or to remove an unwanted consequence, the classical approach to repairing an ontology is deleting enough axioms.
- More fine-grained repairs can be obtained by *weakening axioms* instead of removing them completely.
- In a nutshell (specifically for repairing $\mathcal{EL}$ TBoxes with the particular(!) weakening relation $\succ^{\text{sub}}$):

  Let $\mathcal{T}$ be a TBox and α an unwanted consequence of $\mathcal{T}$.

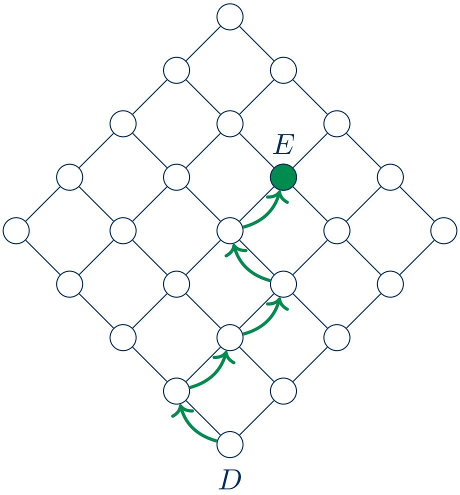
  While there is a justification $\mathcal{J}$ for the unwanted consequence α:

  1. Choose some $C \sqsubseteq D$ in $\mathcal{J}$.
  2. Find a (maximally strong) weakening $C \sqsubseteq E$ where $D \sqsubset_\emptyset E$ and such that $(\mathcal{J} \setminus \{C \sqsubseteq D\}) \cup \{C \sqsubseteq E\}$ does not entail α.
  3. Replace $C \sqsubseteq D$ with $C \sqsubseteq E$ in $\mathcal{T}$.

The $\mathcal{EL}$ Subsumption Hierarchy

Example: Ontology Repair

- **Goal:** Find E such that $C \sqsubseteq E$ is a maximally strong weakening of $C \sqsubseteq D$.
- One might start with D and subsequently go to an upper neighbor until a suitable E has been found.
- Again, we will see later why this will not work in applications.



The $\mathcal{EL}$ Subsumption Hierarchy

Upper Neighbors and Lower Neighbors

- Each concept C has linearly many upper neighbors (modulo equivalence).
- The set of all upper neighbors of a concept C can be computed in polynomial time (modulo equivalence).

The $\mathcal{EL}$ Subsumption Hierarchy
Upper Neighbors and Lower Neighbors

- Each concept C has linearly many upper neighbors (modulo equivalence).
- The set of all upper neighbors of a concept C can be computed in polynomial time (modulo equivalence).

- Each concept C has at most exponentially many lower neighbors (modulo equivalence).
- The set of all lower neighbors of a concept C can be computed in exponential time (modulo equivalence).

The $\mathcal{EL}$ Subsumption Hierarchy

Upper Neighbors and Lower Neighbors

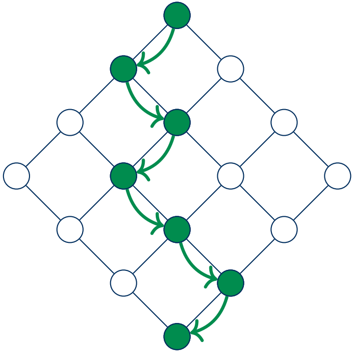
- Each concept C has linearly many upper neighbors (modulo equivalence).
- The set of all upper neighbors of a concept C can be computed in polynomial time (modulo equivalence).

- Each concept C has at most exponentially many lower neighbors (modulo equivalence).
- The set of all lower neighbors of a concept C can be computed in exponential time (modulo equivalence).

- See my paper for further details…

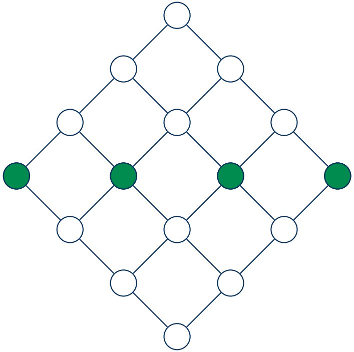
Very Long Chains of Neighbors

- A *chain* is a set of concepts $\{C_1, \ldots, C_\ell\}$ where $C_1 \sqsubset_\emptyset C_2 \sqsubset_\emptyset \cdots \sqsubset_\emptyset C_\ell$.
- **Question:** How long can a chain between two $\mathcal{EL}$ concepts be?



Very Long Chains of Neighbors

- A *chain* is a set of concepts $\{C_1, \ldots, C_\ell\}$ where $C_1 \sqsubset_\emptyset C_2 \sqsubset_\emptyset \cdots \sqsubset_\emptyset C_\ell$.
- **Question:** How long can a chain between two $\mathcal{EL}$ concepts be?
- An *antichain* is a set of concepts $\{D_1, \ldots, D_\ell\}$ where $D_i \not\sqsubseteq_\emptyset D_j$ for all $i \neq j$.
- Why do we need antichains to analyze chains?



Very Long Chains of Neighbors

Let us first fix some notations.

- Assume that $A_1, \ldots, A_k$ are different concept names.
- Further let r_i be a role name for each $i \geq 1$.
- Let $\mathbb{E}_n$ be the part of the $\mathcal{EL}$ subsumption hierarchy consisting of all concepts with a role depth $\leq n$.

Very Long Chains of Neighbors

Let us first fix some notations.

- Assume that $A_1, \ldots, A_k$ are different concept names.
- Further let r_i be a role name for each $i \geq 1$.
- Let $\mathbb{E}_n$ be the part of the $\mathcal{EL}$ subsumption hierarchy consisting of all concepts with a role depth $\leq n$.

We can build chains from antichains as follows.

- $\mathbf{C}_0 \coloneqq \{\, \bigsqcap_{i=1}^{j} A_i \mid 1 \leq j \leq k \,\}$ is a chain in $\mathbb{E}_0$ such that $|\mathbf{C}_0| = k$.
- If $\mathbf{A}_n = \{D_1, \ldots, D_m\}$ is an antichain in $\mathbb{E}_n$, then $\mathbf{C}_{n+1} \coloneqq \{\, \bigsqcap_{i=1}^{j} \exists r_{n+1}.D_i \mid 1 \leq j \leq m \,\}$ is a chain in $\mathbb{E}_{n+1}$ such that $|\mathbf{C}_{n+1}| = |\mathbf{A}_n|$.

Very Long Chains of Neighbors

Further notations:

- For each set $\mathbf{A}$, let $F(\mathbf{A})$ be the set of all sets that consist of exactly half of the elements in $\mathbf{A}$.
- $|F(\mathbf{A})| = f(|\mathbf{A}|)$ where $f(m) \coloneqq \binom{m}{\lfloor \frac{m}{2} \rfloor}$.

Very Long Chains of Neighbors

Further notations:

- For each set $\mathbf{A}$, let $F(\mathbf{A})$ be the set of all sets that consist of exactly half of the elements in $\mathbf{A}$.
- $|F(\mathbf{A})| = f(|\mathbf{A}|)$ where $f(m) \coloneqq \binom{m}{\lfloor \frac{m}{2} \rfloor}$.

We can construct the following antichains.

- $\mathbf{A}_0 \coloneqq \{\, \bigsqcap_{C \in \mathbf{X}} C \mid \mathbf{X} \in F(\{A_1, \ldots, A_k\}) \,\}$ is an antichain in $\mathbb{E}_0$ such that $|\mathbf{A}_0| = f(k)$.
- If $\mathbf{A}_n$ is an antichain in $\mathbb{E}_n$,
  then $\mathbf{A}_{n+1} \coloneqq \{\, \bigsqcap_{C \in \mathbf{X}} \exists r_{n+1}.C \mid \mathbf{X} \in F(\mathbf{A}_n) \,\}$ is an antichain in $\mathbb{E}_{n+1}$ s.t. $|\mathbf{A}_{n+1}| = f(|\mathbf{A}_n|)$.
- **Corollary:** $|\mathbf{A}_n| = \underbrace{f(f(\cdots f}_{n+1 \text{ times}}(k)\cdots)) = f^{n+1}(k)$.

Very Long Chains of Neighbors

Coming back to the chains:

- Since the antichain $\mathbf{A}_{n-1}$ induces the chain $\mathbf{C}_n$ where $|\mathbf{C}_n| = |\mathbf{A}_{n-1}|$ and since $|\mathbf{A}_{n-1}| = f^n(k)$, we obtain:
- **Corollary:** $|\mathbf{C}_n| = f^n(k)$.

Very Long Chains of Neighbors

Coming back to the chains:

- Since the antichain $\mathbf{A}_{n-1}$ induces the chain $\mathbf{C}_n$ where $|\mathbf{C}_n| = |\mathbf{A}_{n-1}|$ and since $|\mathbf{A}_{n-1}| = f^n(k)$, we obtain:
- **Corollary:** $|\mathbf{C}_n| = f^n(k)$.

What's more...

- $\{\exists r_n. \cdots \exists r_1.(A_1 \sqcap \cdots \sqcap A_k)\} \cup \mathbf{C}_n \cup \{\top\}$ is a chain from $\exists r_n. \cdots \exists r_1.(A_1 \sqcap \cdots \sqcap A_k)$ to $\top$ in $\mathbb{E}_n$ with length $\geq f^n(k)$.
- Since each chain can be refined to a chain of neighbors, it follows that there is a chain of neighbors from $\exists r_n. \cdots \exists r_1.(A_1 \sqcap \cdots \sqcap A_k)$ to $\top$ in $\mathbb{E}_n$ with length $\geq f^n(k)$.
- For each two concepts C and D, all chains of neighbors from C to D have the same length.
- Let's take a look on the values $f^n(k)$...

Very Long Chains of Neighbors

The Cases where $k \leq 2$

- $f^n(k) \leq 2$ for each $k \leq 2$ and for each $n \geq 0$.
- Thus: we only get a constant lower bound.
- In fact, for each $k \leq 2$, each chain of neighbors from $\exists r_n. \cdots \exists r_1.(A_1 \sqcap \cdots \sqcap A_k)$ to $\top$ has a length linear in n.

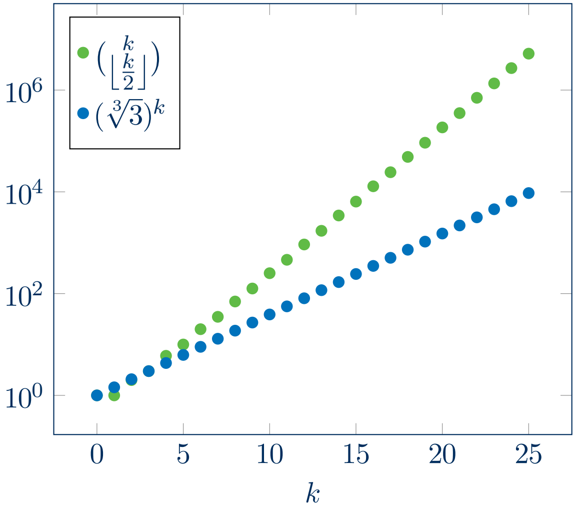
Very Long Chains of Neighbors

The Cases where $k \geq 3$

- $f(k) \geq (\sqrt[3]{3})^k$ for each $k \geq 3$.
- Thus: we get a multi-exponential lower bound, namely with $b \coloneqq \sqrt[3]{3} \approx 1.44$ we have

$$f^n(k) = \underbrace{f(f(\cdots f(k)\cdots))}_{n \text{ times}} \geq \underbrace{b^{b^{\cdot^{\cdot^{\cdot^{b^{b^k}}}}}}}_{n \text{ times}}.$$

- **Corollary:** For each $k \geq 3$, each chain of neighbors from $\exists r_n.\cdots\exists r_1.(A_1 \sqcap \cdots \sqcap A_k)$ to $\top$ has a length n-fold exponential in k.



Very Long Chains of Neighbors

The below table shows some values of the distance from $\exists r_1. \cdots \exists r_n.(A_1 \sqcap \cdots \sqcap A_k)$ to $\top$.

$k \backslash n$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	7
2	2	4	6	8	10	12	14
3	3	$8 \geq \binom{3}{1}$	$20 \geq 4$	$84 \geq \binom{4}{2}$	$8573 \geq \binom{6}{3}$	$? \geq \binom{20}{10}$	$? \geq \binom{184756}{92378}$
4	4	$16 \geq \binom{4}{2}$	$168 \geq \binom{6}{3}$	$? \geq \binom{20}{10}$	$? \geq \binom{184756}{92378}$	$? \gtrapprox \binom{2.33 \cdot 10^{55614}}{1.16 \cdot 10^{55614}}$	$?$
5	5	$32 \geq \binom{5}{2}$	$7581 \geq \binom{10}{5}$	$? \geq \binom{252}{126}$	$? \gtrapprox \binom{3.63 \cdot 10^{74}}{1.82 \cdot 10^{74}}$	$?$	$?$
6	6	$64 \geq \binom{6}{3}$	$? \geq \binom{20}{10}$	$? \geq \binom{184756}{92378}$	$? \gtrapprox \binom{2.33 \cdot 10^{55614}}{1.16 \cdot 10^{55614}}$	$?$	$?$
7	7	$128 \geq \binom{7}{3}$	$? \geq \binom{35}{17}$	$? \gtrapprox \binom{4.54 \cdot 10^{9}}{2.27 \cdot 10^{9}}$	$?$	$?$	$?$
8	8	$256 \geq \binom{8}{4}$	$? \geq \binom{70}{35}$	$? \gtrapprox \binom{1.12 \cdot 10^{20}}{5.61 \cdot 10^{19}}$	$?$	$?$	$?$
9	9	$512 \geq \binom{9}{4}$	$? \geq \binom{126}{63}$	$? \gtrapprox \binom{6.03 \cdot 10^{36}}{3.02 \cdot 10^{36}}$	$?$	$?$	$?$
10	10	$1024 \geq \binom{10}{5}$	$? \geq \binom{252}{126}$	$? \gtrapprox \binom{3.63 \cdot 10^{74}}{1.82 \cdot 10^{74}}$	$?$	$?$	$?$

A Consequence and an Application

Coming Back to the two Initial Examples: Concept Learning and Ontology Repair

- **Corollary:** Due to the existence of very long chains, one should never try to find a target concept by going along the neighborhood relation only without making jumps.
- For $\mathcal{EL}$, it can be shown that an **ideal** upward refinement operator applied to a concept C yields exactly the set of upper neighbors of C, and dually an **ideal** downward refinement operator applied to a concept C yields the set of lower neighbors of C.
- Thus, if one wants to utilize refinement operators in $\mathcal{EL}$ and cannot bound the number of consecutive refinement steps, one should not try to use an **ideal** refinement operator in applications.

A Consequence and an Application

Deciding Optimality

- After all, we can devise a useful application…
- Sometimes, one wants to check whether a concept C is maximally specific for a monotonic property $\mathcal{P}$.
- To do so, one might enumerate the lower neighbors of C and then test whether any of these satisfies $\mathcal{P}$.
- This works dually with upper neighbors for checking if C is maximally general for $\mathcal{P}$.

That’s it for now!

Do you have questions or comments?