Research Challenges
Challenges
Challenges

- Increased expressive power
  - Existing DL systems implement (at most) $SHIN/SHIQ$
  - OWL extends $SHIN$ with datatypes (Lite) and nominals (DL)
Challenges

- **Increased expressive power**
  - Existing DL systems implement (at most) $SHIN/SHIQ$
  - OWL extends $SHIN$ with datatypes (Lite) and nominals (DL)

- **Scalability**
  - Very large KBs
  - Reasoning with (very large numbers of) individuals
Challenges

**Increased expressive power**
- Existing DL systems implement (at most) $SHIN/SHIQ$
- OWL extends $SHIN$ with datatypes (Lite) and nominals (DL)

**Scalability**
- Very large KBs
- Reasoning with (very large numbers of) individuals

**Other reasoning tasks**
- Querying
- Matching
- Least common subsumer
- ...
Challenges

- **Increased expressive power**
  - Existing DL systems implement (at most) $SHIN/SHIQ$
  - OWL extends $SHIN$ with datatypes (Lite) and nominals (DL)

- **Scalability**
  - Very large KBs
  - Reasoning with (very large numbers of) individuals

- **Other reasoning tasks**
  - Querying
  - Matching
  - Least common subsumer
  - ...

- **Tools and Infrastructure**
  - Support for large scale ontological engineering and deployment
Increased Expressive Power: Datatypes

- **OWL** has simple form of datatypes
- **Unary predicates plus disjoint object-class/datatype domains**
- **Well understood theoretically**
- **Existing work on concrete domains** [Baader & Hanschke, Lutz]
- **Algorithm already known for** SHOQ (D) [Horrocks & Sattler]
- Can use **hybrid reasoning** (DL reasoner + datatype "oracle")
- **May be practically challenging**
- **Large number of XMLS datatypes may need to be supported**
- **Already seeing some (partial) implementations** Cerebra system (Network Inference), Racer system (Hamburg)
Increased Expressive Power: Datatypes

- **OWL** has simple form of datatypes
  - Unary predicates plus disjoint object-class/datatype domains

- Well understood theoretically
- Existing work on concrete domains [Baader & Hanschke, Lutz]
- Algorithm already known for SHOQ (D) [Horrocks & Sattler]
- Can use hybrid reasoning (DL reasoner + datatype "oracle")
- May be practically challenging
  - Large number of XML datatypes may need to be supported (?)
- Already seeing some (partial) implementations
  - Cerebra system (Network Inference), Racer system (Hamburg)
Increased Expressive Power: Datatypes

OWL has simple form of datatypes
  ● Unary predicates plus disjoint object-class/datatype domains

Well understood theoretically
  ● Existing work on concrete domains [Baader & Hanschke, Lutz]
  ● Algorithm already known for \textit{SHOQ(D)} [Horrocks & Sattler]
  ● Can use \textit{hybrid reasoning} (DL reasoner + datatype “oracle”)

Reasoning with Expressive Description Logics – p. 3/8
Increased Expressive Power: Datatypes

OWL has simple form of datatypes
- Unary predicates plus disjoint object-class/datatype domains

Well understood theoretically
- Existing work on concrete domains [Baader & Hanschke, Lutz]
- Algorithm already known for $SHOQ(D)$ [Horrocks & Sattler]
- Can use hybrid reasoning (DL reasoner + datatype “oracle”)

May be practically challenging
- Large number of XMLS datatypes may need to be supported (?)
**Increased Expressive Power: Datatypes**

- **OWL** has simple form of datatypes
  - Unary predicates plus disjoint object-class/datatype domains
  - Well understood *theoretically*
    - Existing work on **concrete domains** [Baader & Hanschke, Lutz]
    - Algorithm already known for $SHOQ(D)$ [Horrocks & Sattler]
    - Can use **hybrid reasoning** (DL reasoner + datatype “oracle”)
  - May be **practically** challenging
    - Large number of XMLS datatypes may need to be supported (?)
  - Already seeing some (partial) **implementations**
    - Cerebra system (Network Inference), Racer system (Hamburg)
Increased Expressive Power: Nominals

- Extensionally defined concepts, e.g., EU, France, Italy, ... (g)

- Theoretically very challenging
- Resulting logic has known high complexity (NExpTime)
- No known "practical" algorithm
- Not obvious how to extend tableaux techniques in this direction
  - Loss of tree model property
  - Spy-points: $\forall v \exists R: f \gSpy_g v \in R$
  - Finite domains: $\forall \Spy \exists v \in R$

- Standard solution is weaker semantics for nominals
- Treat nominals as primitive classes
- Loss of completeness/soundness
Increased Expressive Power: Nominals

OWL DL `oneOf` constructor equivalent to hybrid logic nominals
- Extensionally defined concepts, e.g., $\text{EU} \equiv \{\text{France, Italy, } \ldots\}$
Increased Expressive Power: Nominals

OWL DL oneOf constructor equivalent to hybrid logic nominals

- Extensionally defined concepts, e.g., EU ≡ \{France, Italy, \ldots\}

Theoretically very challenging

- Resulting logic has known high complexity (NExpTime)
- No known “practical” algorithm
- Not obvious how to extend tableaux techniques in this direction
  - Loss of tree model property
  - Spy-points: $\top \sqsubseteq \exists R.\{Spy\}$
  - Finite domains: $\{Spy\} \sqsubseteq \leq nR^-$
Increased Expressive Power: Nominals

- OWL DL `oneOf` constructor equivalent to hybrid logic nominals
  - Extensionally defined concepts, e.g., \( \text{EU} \equiv \{ \text{France, Italy, \ldots} \} \)

- Theoretically very challenging
  - Resulting logic has known high complexity (NExpTime)
  - No known “practical” algorithm
  - Not obvious how to extend tableaux techniques in this direction
    - Loss of tree model property
    - Spy-points: \( \top \subseteq \exists R. \{ \text{Spy} \} \)
    - Finite domains: \( \{ \text{Spy} \} \subseteq \leq n R^- \)

- Standard solution is weaker semantics for nominals
  - Treat nominals as primitive classes
  - Loss of completeness/soundness
Increased Expressive Power: Extensions

OWL not expressive enough for all applications
Increased Expressive Power: Extensions

OWL **not expressive enough** for all applications

Extensions **wish list** includes:

- Feature chain (path) agreement, e.g., output of component of composite process equals input of subsequent process
- Complex roles/role inclusions, e.g., a city located in part of a country is located in that country
- Rules—proposal(s) already exist for “LP style rules”
- Temporal and spatial reasoning
- ...
Increased Expressive Power: Extensions

- **OWL not expressive enough** for all applications

- Extensions **wish list** includes:
  - Feature chain (path) agreement, e.g., output of component of composite process equals input of subsequent process
  - Complex roles/role inclusions, e.g., a city located in part of a country is located in that country
  - Rules—proposal(s) already exist for “LP style rules”
  - Temporal and spatial reasoning
  - ...

- May be impossible/undesirable to resist such extensions
Increased Expressive Power: Extensions

- OWL not expressive enough for all applications
- Extensions wish list includes:
  - Feature chain (path) agreement, e.g., output of component of composite process equals input of subsequent process
  - Complex roles/role inclusions, e.g., a city located in part of a country is located in that country
  - Rules—proposal(s) already exist for “LP style rules”
  - Temporal and spatial reasoning
  - ...
- May be impossible/undesirable to resist such extensions
- Extended language sure to be undecidable
Increased Expressive Power: Extensions

- OWL not expressive enough for all applications
- Extensions wish list includes:
  - Feature chain (path) agreement, e.g., output of component of composite process equals input of subsequent process
  - Complex roles/role inclusions, e.g., a city located in part of a country is located in that country
  - Rules—proposal(s) already exist for “LP style rules”
  - Temporal and spatial reasoning
  - . . .
- May be impossible/undesirable to resist such extensions
- Extended language sure to be undecidable
- How can extensions best be integrated with OWL?
Increased Expressive Power: Extensions

- OWL not expressive enough for all applications
- Extensions wish list includes:
  - Feature chain (path) agreement, e.g., output of component of composite process equals input of subsequent process
  - Complex roles/role inclusions, e.g., a city located in part of a country is located in that country
  - Rules—proposal(s) already exist for “LP style rules”
  - Temporal and spatial reasoning
  - ...
- May be impossible/undesirable to resist such extensions
- Extended language sure to be undecidable
- How can extensions best be integrated with OWL?
- How can reasoners be developed/adapted for extended languages
  - Some existing work on language fusions [Baader et al] and hybrid reasoners
Scalability

Reasoning hard (ExpTime) even without nominals (i.e., SHIN)

Web ontologies may grow very large

Good empirical evidence of scalability/tractability for DL systems

E.g., 5,000 (complex) classes; 100,000+ (simple) classes

But evidence mostly w.r.t. SHF (no inverse)

Problems can arise when SHF extended to SHIN

Important optimisations no longer (fully) work

Reasoning with individuals

Deployment of web ontologies will mean reasoning with (possibly very large numbers of) individuals/tuples

Unlikely that standard Abox techniques will be able to cope
Scalability

- Reasoning hard (ExpTime) even without nominals (i.e., $SHIN$)

Web ontologies may grow very large. Good empirical evidence of scalability/tractability for DL systems. E.g., 5,000 (complex) classes; 100,000+ (simple) classes. But evidence mostly w.r.t. $SHF$ (no inverse). Problems can arise when $SHF$ extended to $SHIN$. Important optimisations no longer (fully) work.

Deployment of web ontologies will mean reasoning with (possibly very large numbers of) individuals/tuples. Unlikely that standard Abox techniques will be able to cope.
Scalability

- Reasoning hard (ExpTime) even without nominals (i.e., $SHIN$)
- Web ontologies may grow very large
Scalability

- Reasoning **hard** (ExpTime) even without nominals (i.e., $SHIN$)
- Web ontologies may grow **very large**
- Good **empirical evidence** of scalability/tractability for DL systems
  - E.g., 5,000 (complex) classes; 100,000+ (simple) classes
Scalability

- Reasoning hard (ExpTime) even without nominals (i.e., $SHIN$)
- Web ontologies may grow very large
- Good empirical evidence of scalability/tractability for DL systems
  - E.g., 5,000 (complex) classes; 100,000+ (simple) classes
- But evidence mostly w.r.t. $SHF$ (no inverse)
Scalability

- Reasoning **hard** (ExpTime) even without nominals (i.e., $SHIN$)
- Web ontologies may grow **very large**
- Good **empirical evidence** of scalability/tractability for DL systems
  - E.g., 5,000 (complex) classes; 100,000+ (simple) classes
- But evidence mostly w.r.t. $SHF$ (no inverse)
- **Problems** can arise when $SHF$ extended to $SHIN$
  - Important **optimisations** no longer (fully) work
Scalability

- Reasoning **hard** (ExpTime) even without nominals (i.e., $SHIN$)
- Web ontologies may grow **very large**
- Good **empirical evidence** of scalability/tractability for DL systems
  - E.g., 5,000 (complex) classes; 100,000+ (simple) classes
- But evidence mostly w.r.t. $SHF$ (no inverse)
- **Problems** can arise when $SHF$ extended to $SHIN$
  - Important **optimisations** no longer (fully) work
- Reasoning with **individuals**
  - **Deployment** of web ontologies will mean reasoning with (possibly very large numbers of) individuals/tuples
  - Unlikely that standard **Abox** techniques will be able to cope
Performance Solutions (Maybe)

Excessive memory usage
Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)
Promising results from more precise blocking condition [Sattler & Horrocks]

Caching and merging can still work in some situations (work in progress)

Reasoning with very large KBs
DL systems shown to work with 100k concept KB [Haarslev & Möller]
But KB only exploited small part of DL language
Performance Solutions (Maybe)

Excessive memory usage

Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)

Promising results from more precise blocking condition [Sattler & Horrocks]

Caching and merging can still work in some situations (work in progress)

Reasoning with very large KBs

DL systems shown to work with 100k concept KB [Haarslev & Möller]

But KB only exploited small part of DL language
Performance Solutions (Maybe)

- Excessive memory usage
  - Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)
  - Promising results from more precise blocking condition [Sattler & Horrocks]

Reasoning with Expressive Description Logics – p. 7/8
Performance Solutions (Maybe)

- Excessive **memory usage**
  - Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)
  - Promising results from more precise blocking condition [Sattler & Horrocks]

- **Caching** and merging
Performance Solutions (Maybe)

- Excessive **memory usage**
  - Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)
  - Promising results from more precise blocking condition [Sattler & Horrocks]

- **Caching** and merging
  - Can still work in some situations (work in progress)
Performance Solutions (Maybe)

- Excessive memory usage
  - Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)
  - Promising results from more precise blocking condition [Sattler & Horrocks]

- Caching and merging
  - Can still work in some situations (work in progress)

- Reasoning with very large KBs
Performance Solutions (Maybe)

- Excessive **memory usage**
  - Problem exacerbated by over-cautious double blocking condition (e.g., root node can never block)
  - Promising results from more precise blocking condition [Sattler & Horrocks]

- **Caching** and merging
  - Can still work in some situations (work in progress)

- Reasoning with **very large KBs**
  - DL systems shown to work with $\approx 100k$ concept KB [Haarslev & Möller]
  - But KB only exploited small part of DL language
Other Reasoning Tasks

Querying
Retrieval and instantiation won't be sufficient. Minimum requirement will be a DB style query language. May also need "what can I say about x?" style of query.

Explanation
To support ontology design, justifications and proofs (e.g., of query results).

"Non-Standard Inferences", e.g., LCS, matching
To support ontology integration and "bottom-up" design of ontologies.
Other Reasoning Tasks

Querying

- Retrieval and instantiation won't be sufficient
- Minimum requirement will be **DB style query language**
- May also need “what can I say about \( x \)” style of query
Other Reasoning Tasks

- **Querying**
  - Retrieval and instantiation won't be sufficient
  - Minimum requirement will be **DB style query language**
  - May also need “what can I say about $x$?” style of query

- **Explanation**
  - To support ontology design
  - Justifications and proofs (e.g., of query results)
Other Reasoning Tasks

Querying
- Retrieval and instantiation won't be sufficient
- Minimum requirement will be **DB style query language**
- May also need "what can I say about $x$?" style of query

Explanation
- To support ontology design
- Justifications and proofs (e.g., of query results)

"Non-Standard Inferences", e.g., LCS, matching
- To support ontology integration
- To support "bottom up" design of ontologies