A Survey of Temporal Knowledge Representations

Sami Al-Dhaheri

Advisor: Professor Abdullah Tansel

Second Exam Presentation
Knowledge Representations

- **logic-based formalisms**
  - more complex and difficult to understand
  - all based on formal logic
    - Syntax
    - Semantics
  - E.g.:
    - Description Logics

- **non logic-based formalisms**
  - closer to human intuition
  - therefore easier to understand
  - usually don’t have consistent semantics
    - E.g.:
      - Semantic Networks
      - Frame-based representations
      - Rule-based representations
FOL as Semantic Web Language?

• Why not simply take FOL for Ontologies?
• FOL can do everything..
  • *compare higher programming Languages to assemblers*
    • FOL has
      • high expressivity
      • too bulky for modelling
      • not appropriate to find consensus in modelling
      • proof theoretically very complex (semi-decidable)
  • FOL is also not a markup language

Look for an appropriate fragment of FOL
Description Logics (DLs)

• DLs are Fragments of FOL
• In DL from simple descriptions more complex descriptions are created with the help of Constructors.
• DLs differ in the applied constructors (Expressivity)
• DLs have been developed from “semantic Networks“
• DLs are decidable (most times)
• DLs possess sufficient expressivity (most times)
• DLs are related to modal logics
• e.g., W3C Standard OWL 2 DL is based on description logics $SHROIQ(D)$
Attributive Language with complements ALC

Building Blocks:
- Concepts/ Classes
- Roles / Properties
- Individuals

- **Student (John)**
  Individual John is of class Student

- **Lecture (Database)**
  Individual Database is of class lecture

- **attendsLecture (John, Database)**
  John attends the lecture Database
**ALC - Building Blocks**

- **Atomic Types**
  - Concept names \( A, B, \ldots \)
  - Special concepts
    - \( \top \) - Top (universal concept)
    - \( \bot \) - Bottom concept
  - Role names \( R, S, \ldots \)

- **Constructors**
  - Negation: \( \neg C \)
  - Conjunction: \( C \sqcap D \)
  - Disjunction: \( C \sqcup D \)
  - Existential quantifier: \( \exists R.C \)
  - Universal quantifier: \( \forall R.C \)
ALC - Building Blocks

• Class Inclusion
  • Professor ⊑ FacultyMember
  • every Professor is a Faculty Member
  • In FOL equals $(\forall x)(\text{Professor}(x) \rightarrow \text{FacultyMember}(x))$

• Class Equivalence
  • Professor ≡ FacultyMember
  • the Faculty Members are exactly the Professors
  • In FOL equals $(\forall x)(\text{Professor}(x) \leftrightarrow \text{FacultyMember}(x))$
ALC - Complex Class Relations

- Conjunction $\sqcap$
- Disjunction $\sqcup$
- Negation $\neg$

Professor $\equiv (\text{Person} \sqcap \text{UniversityEmployee})$
$\sqcup (\text{Person} \sqcap \neg \text{Student})$

$(\forall x)(\text{Professor}(x) \rightarrow ((\text{Person}(x) \land \text{UniversityEmployee}(x)))$
$\lor (\text{Person}(x) \land \neg \text{Student}(x)))$
ALC - Quantifiers on Roles

• **Strict Binding** of the Range of a Role to a Class
  • Examination $\equiv \forall \text{hasSupervisor}.\text{Professor}$
  • An Examination must be supervised by a Professor
  \[(\forall x)(\text{Examination}(x) \rightarrow (\forall y)(\text{hasSupervisor}(x,y) \rightarrow \text{Professor}(y)))\]

• **Open Binding** of the Range of a Role to a Class
  • Examination $\equiv \exists \text{hasSupervisor}.\text{Person}$
  • Every Examination has at least one supervisor (who is a person)
  \[(\forall x)(\text{Examination}(x) \rightarrow (\exists y)(\text{hasSupervisor}(x,y) \land \text{Person}(y)))\]
**ALC - Formal Syntax**

- Production rules for creating classes in ALC:

  \( A \) is an atomic class, \( C \) and \( D \) are complex classes and \( R \) is a Role

\[
\begin{align*}
  C, D &= A | T | ⊥ | ¬C | C \sqcap D | C \sqcup D | \exists R.C | \forall R.C \\
\end{align*}
\]
**ALC - Formal Syntax**

- Production rules for creating classes in ALC:

\[(A \text{ is an atomic class, } C \text{ and } D \text{ are complex classes and } R \text{ is a Role})\]

\[
\begin{align*}
C, D & := A \mid \top \mid \bot \mid \neg C \mid C \land D \mid C \lor D \mid \exists R.C \mid \forall R.C \\
\end{align*}
\]

- An ALC **TBox** contains assertions of the form \(C \sqsubseteq D\) and \(C \equiv D\), where \(C, D\) are complex classes.
**ALC - Formal Syntax**

• Production rules for creating classes in ALC:

\[
(A \text{ is an atomic class, } C \text{ and } D \text{ are complex classes and } R \text{ is a Role})
\]

\[
C, D := A | T | \bot | \neg C | C \sqcap D | C \sqcup D | \exists R.C | \forall R.C
\]

• An ALC **TBox** contains assertions of the form \( C \sqsubseteq D \) and \( C \equiv D \), where \( C, D \) are complex classes.

• An ALC **ABox** contains assertions of the form \( C(a) \) and \( R(a, b) \), where \( C \) is a complex Class, \( R \) a Role and \( a, b \) Individuals.
ALC - Formal Syntax

• Production rules for creating classes in ALC:

\[(A \text{ is an atomic class, } C \text{ and } D \text{ are complex classes and } R \text{ is a Role})\]

\[C, D := A | T | \bot | \neg C | C \cap D | C \cup D | \exists R.C | \forall R.C\]

• An ALC TBox contains assertions of the form \(C \sqsubseteq D\) and \(C \equiv D\), where \(C, D\) are complex classes.

• An ALC ABox contains assertions of the form \(C(a)\) and \(R(a, b)\), where \(C\) is a complex Class, \(R\) a Role and \(a, b\) Individuals.

• An ALC-Knowledge Base contains an ABox and a TBox.
**ALC - Semantic (Interpretation)**

• we define a model-theoretic semantic for ALC (i.e. Entailment will be defined via Interpretations)

• an **Interpretation** $I = (\Delta^I, .^I)$ contains

• a set $\Delta^I$ (Domain) of Individuals and

• an **interpretation function** $.^I$ that maps
  
  • Individual names $a$
    
    to domain elements $a^I \in \Delta^I$

  • Class names $C$
    
    to a set of domain elements $C^I \subseteq \Delta^I$

  • Role names $R$
    
    to a set of pairs of domain elements $R^I \subseteq \Delta^I \times \Delta^I$
ALC - Semantic (Interpretation)
ALC - Semantic (Interpretation)

• Extension for complex classes:

\[
\top^I = \Delta^I \quad \text{and} \quad \bot^I = \emptyset
\]
**ALC - Semantic (Interpretation)**

- Extension for complex classes:

\[
\begin{align*}
T^I &= \Delta^I \quad \text{and} \quad \bot^I = \emptyset \\
(C \cup D)^I &= C^I \cup D^I \quad \text{and} \\
(C \cap D)^I &= C^I \cap D^I
\end{align*}
\]
**ALC - Semantic (Interpretation)**

- Extension for complex classes:

- $T^I = \Delta^I$ and $\bot^I = \emptyset$
- $(C \cup D)^I = C^I \cup D^I$ and $(C \cap D)^I = C^I \cap D^I$
- $(\neg C)^I = \Delta^I \setminus C^I$
ALC - Semantic (Interpretation)

• Extension for complex classes:

\[ T^I = \Delta^I \quad \text{and} \quad \bot^I = \emptyset \]

\[ (C \cup D)^I = C^I \cup D^I \quad \text{and} \]
\[ (C \cap D)^I = C^I \cap D^I \]

\[ (\neg C)^I = \Delta^I \setminus C^I \]

\[ \forall R. C = \{ a \in \Delta^I | (\forall b \in \Delta^I) \left( (a, b) \in R^I \to b \in C^I \right) \} \]
ALC - Semantic (Interpretation)

• Extension for complex classes:

\[ T^I = \Delta^I \quad \text{and} \quad \bot^I = \emptyset \]

\[ (C \cup D)^I = C^I \cup D^I \quad \text{and} \quad (C \cap D)^I = C^I \cap D^I \]

\[ (\neg C)^I = \Delta^I \setminus C^I \]

\[ \forall R. C = \{ a \in \Delta^I | (\forall b \in \Delta^I) ((a, b) \in R^I \rightarrow b \in C^I) \} \]

\[ \exists R. C = \{ a \in \Delta^I | (\exists b \in \Delta^I) ((a, b) \in R^I \land b \in C^I) \} \]
...and Axioms:

- **C(a)** holds, iff \( a^I \in C^I \)
- **R(a, b)** holds, iff \( (a^I, b^I) \in R^I \)
- **C ⊆ D** holds, iff \( C^I \subseteq D^I \)
- **C ≡ D** holds, iff \( C^I = D^I \)
ALC - Knowledgebase

• Terminological Knowledge (TBox)
  Axioms that describe the structure of the modeled domain (conceptual schema):
  • Human ⊑ ∃parentOf.Human
  • Orphan ≡ Human △ ¬∃hasParent.Alive

• Assertional Knowledge (ABox)
  Axioms that describe specific situations (data):
  • Orphan(harrypotter)
  • hasParent(harrypotter, jamespotter)
Tableaux Algorithm:

• Proof algorithm to check the consistency of a logical formula by inferring that its negation is a contradiction (proof by refutation).

• Construct tree, where each node is marked with a logical formula. A path from the root to a leaf is the conjunction of all formulas represented within the nodes of the path; a branch of the path represents a disjunction.

• The tree is created by successive application of the Tableaux Extension Rules.
• Transformation to *Negation normalform* necessary

• Let \( W \) be a knowledge base,
  • Substitute \( C \equiv D \) by \( C \sqsubseteq D \) and \( D \sqsubseteq C \)
  • Substitute \( C \sqsubseteq D \) by \( \neg C \sqcup D \).

• Apply the NNF Transformations from the next page

• Resulting knowledge base \( \text{NNF}(W) \)
  • Negation normalform of \( W \).
  • Negation is placed directly in front of atomic classes.
Tableaux Transformation in Negation Normalform

• NNF Transformations

NNF(C) = C, if C is atomic
NNF(¬C) = ¬C, if C is atomic
NNF(¬¬C) = C
NNF(C ⊔ D) = NNF(C) ⊔ NNF(D)
NNF(C ⊓ D) = NNF(C) ⊓ NNF(D)
NNF(¬(C ⊔ D)) = NNF(¬C) ⊓ NNF(¬D)
NNF(¬(C ⊓ D)) = NNF(¬C) ⊔ NNF(¬D)
NNF(∀R.C) = ∀R.NNF(C)
NNF(∃R.C) = ∃R.NNF(C)
NNF(¬∀R.C) = ∃R.NNF(¬C)
NNF(¬∃R.C) = ∀R.NNF(¬C)

• W and NNF(W) are logically equivalent.
Tableaux Transformation in Negation Normalform

• Example: $\Phi \models (E \cap U) \cup \neg (E \cup D)$

• In NNF: $\neg \Phi \cup (E \cap U) \cup (E \cap \neg D)$

$C \in D = \neg C \cup D$

$\neg (C \cup D) = \neg C \cap \neg D$
**Tableaux Extension Rules for DL**

<table>
<thead>
<tr>
<th>Selection</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C(a) \in W (ABox) )</td>
<td>Add ( C(a) )</td>
</tr>
<tr>
<td>( R(a,b) \in W (ABox) )</td>
<td>Add ( R(a,b) )</td>
</tr>
<tr>
<td>( C \in W (TBox) )</td>
<td>Add ( C(a) ) for a known Individual ( a )</td>
</tr>
<tr>
<td>( (C \sqcap D)(a) \in A )</td>
<td>Add ( C(a) ) and ( D(a) )</td>
</tr>
<tr>
<td>( (C \sqcup D)(a) \in A )</td>
<td>Split the path. Add (1) ( C(a) ) and (2) ( D(a) )</td>
</tr>
<tr>
<td>( (∃R.C)(a) \in A )</td>
<td>Add ( R(a,b) ) and ( C(b) ) for a new Individual ( b )</td>
</tr>
<tr>
<td>( (∀R.C)(a) \in A )</td>
<td>if ( R(a,b) \in A ), then add ( C(b) )</td>
</tr>
</tbody>
</table>

- If the resulting tableaux is closed, the original knowledge base is unsatisfiable.
- Only select elements that lead to new elements within the tableaux. If this is not possible, then the algorithm terminates and the original knowledge base is satisfiable.
Tableaux Algorithm (DL) Example:

- P … Professor
- E … Person
- U … University Employee
- D … Student

- Knowledge Base: $P \sqsubseteq (E \cap U) \sqcup (E \cap \neg D)$
- Is $P \sqsubseteq E$ a logical consequence?

- Knowledge Base (with [negated] query) in NNF:
  \{\neg P \sqcup (E \cap U) \sqcup (E \cap \neg D) , (P \sqcap \neg E) (a) \}
Tableaux Algorithm (DL) Example:

- Knowledge Base $W: \{ \neg P \cup (E \cap U) \cup (E \cap \neg D), (P \cap \neg E)(a) \}$
Tableaux Algorithm (DL) Example:

- Knowledge Base $\mathcal{W}$: $\{ \neg P \cup (E \cap U) \cup (E \cap \neg D), (P \cap \neg E) (a) \}$
- Tableaux $A$: 
Tableaux Algorithm (DL) Example:

- Knowledge Base $W: \{ \neg P \lor (E \land \neg U) \lor (E \land \neg D), (P \land \neg E) (a) \}$
- Tableaux $A$:

  $(1) (P \land \neg E) (a)$ (from knowledge base)
Tableaux Algorithm (DL) Example:

- **Knowledge Base** $W: \{ \neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D), (P \sqcap \neg E)(a) \}$

- **Tableaux** $A$:

  $(1) (P \sqcap \neg E)(a)$ (from knowledge base)

  $(2|\alpha \text{ from } 1) P(a)$
Tableaux Algorithm (DL) Example:

- **Knowledge Base** \( \mathcal{W} : \{ \neg P \cup (E \cap U) \cup (E \cap \neg D), (P \cap \neg E) (a) \} \)
- **Tableaux A:**

  (1) \((P \cap \neg E) (a)\) (from knowledge base)

  (2) \(\alpha \text{ from 1) } P (a)\)

  (3) \(\alpha \text{ from 1) } \neg E (a)\)
Tableaux Algorithm (DL) Example:

- **Knowledge Base** $\mathcal{W}: \{ \neg P \cup (E \cap U) \cup (E \cap \neg D), (P \cap \neg E) (a) \}$

- **Tableaux** $\mathcal{A}$:

  1. $(P \cap \neg E) (a)$ (from knowledge base)
  2. $(\alpha$ from 1) $P (a)$
  3. $(\alpha$ from 1) $\neg E (a)$
  4. $(\neg P \cup (E \cap U) \cup (E \cap \neg D)) (a)$ (from knowledge base)
Tableaux Algorithm (DL) Example:

- **Knowledge Base** $\mathcal{W}$: $\{ \neg P \sqcup (E \cap U) \sqcup (E \cap \neg D), (P \cap \neg E) (a) \}$
- **Tableaux $\mathcal{A}$:**
  1. $(P \cap \neg E) (a)$ (from knowledge base)
  2. $P (a)$ (from 1)
  3. $\neg E (a)$ (from 1)
  4. $(\neg P \sqcup (E \cap U) \sqcup (E \cap \neg D)) (a)$ (from knowledge base)
  5. $\neg P (a)$ (from 4)
  6. $(E \cap U) \sqcup (E \cap \neg D) (a)$
Tableaux Algorithm (DL) Example:

• Knowledge Base $\mathcal{W}: \{-P \lor (E \cap U) \lor (E \cap \neg D), (P \land \neg E)(a)\}$

• Tableaux $A$:

(1) $(P \land \neg E)(a)$ (from knowledge base)

(2) $\alpha$ from 1) $P(a)$

(3) $\alpha$ from 1) $\neg E(a)$

(4) $\neg P \lor (E \cap U) \lor (E \cap \neg D)(a)$ (from knowledge base)

(5) $\beta$ from 4) $\neg P(a)$ | (6) $(E \cap U) \lor (E \cap \neg D)(a)$

(7) $\beta$ from 6) $(E \cap U)(a)$ | (8) $(E \cap \neg D)(a)$
Tableaux Algorithm (DL) Example:

- Knowledge Base $\mathcal{W}: \{ \neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D), (P \sqcap \neg E)(a) \}$
- Tableaux $A$:

$(1) (P \sqcap \neg E)(a) \quad \text{(from knowledge base)}$

$(2|\alpha \text{ from } 1) P(a)$

$(3|\alpha \text{ from } 1) \neg E(a)$

$(4) (\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a) \quad \text{(from knowledge base)}$

$(5|\beta \text{ from } 4)) \neg P(a) \quad | \quad (6) ((E \sqcap U) \sqcup (E \sqcap \neg D))(a)$

$(7|\beta \text{ from } 6) (E \sqcap U)(a) \quad | \quad (8) (E \sqcap \neg D)(a)$

$(9|\alpha \text{ from } 7) E(a)$
Tableaux Algorithm (DL) Example:

- Knowledge Base $\mathcal{W}$: \{\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D), (P \sqcap \neg E)(a) \}

- Tableaux $A$:

  1. $(P \sqcap \neg E)(a)$ (from knowledge base)

     2. $\alpha$ from 1) $P(a)$

     3. $\alpha$ from 1) $\neg E(a)$

     4. $(\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a)$ (from knowledge base)

     5. $\beta$ from 4)) $\neg P(a)$

     6. $(\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D))(a)$

     7. $\beta$ from 6) $(E \sqcap U)(a)$

     8. $(E \sqcap \neg D)(a)$

     9. $\alpha$ from 7) $E(a)$

     10. $\alpha$ from 7) $U(a)$
Tableaux Algorithm (DL) Example:

- **Knowledge Base** \( \mathcal{W} : \neg P \sqcup (E \cap U) \sqcup (E \cap \neg D), (P \sqcap \neg E) (a) \)

- **Tableaux** \( \mathcal{A} : \)

  1. \( (P \sqcap \neg E) (a) \) (from knowledge base)
  
     2. | \( \alpha \) from 1 \( P (a) \)
  
     3. | \( \alpha \) from 1 \( \neg E (a) \)
  
     4. \( (\neg P \sqcup (E \cap U) \sqcup (E \cap \neg D)) (a) \) (from knowledge base)
  
     5. | \( \beta \) from 4 \( \neg P (a) \)
  
     6. \( ((E \cap U) \sqcup (E \cap \neg D)) (a) \)
  
     7. | \( \beta \) from 6 \( (E \cap U) (a) \)
  
     8. \( (E \cap \neg D) (a) \)
  
     9. | \( \alpha \) from 7 \( E (a) \)
  
     10. | \( \alpha \) from 8 \( E (a) \)
  
     11. | \( \alpha \) from 7 \( U (a) \)
Tableaux Algorithm (DL) Example:

- Knowledge Base $W: \{ \neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D), (P \sqcap \neg E) (a) \}$
- Tableaux $A$:

  (1) $(P \sqcap \neg E) (a)$ (from knowledge base)

  (2|α from 1) $P (a)$

  (3|α from 1) $\neg E (a)$

  (4) $(\neg P \sqcup (E \sqcap U) \sqcup (E \sqcap \neg D)) (a)$ (from knowledge base)

  (5|β from 4)) $\neg P (a)$ | (6) $( (E \sqcap U) \sqcup (E \sqcap \neg D) ) (a)$

  (7|β from 6) $(E \sqcap U) (a)$ | (8) $(E \sqcap \neg D) (a)$

  (9|α from 7) $E (a)$  (10|α from 8) $E (a)$

  (11|α from 7) $U (a)$  (12|α from 8) $\neg D (a)$
The Selection of $(\exists R.C)(a)$ in the tableaux path $A$ is blocked, if there is already an individual $b$ with \[ \{ C | C(a) \in A \} \subseteq \{ C | C(b) \in A \}. \]

Two possibilities of termination:

1. Closing the Tableaux.
   Knowledge Base is unsatisfiable.

2. All non blocked selections from the tableaux don’t lead to an extension.
   Knowledge Base is satisfiable.
Temporal extensions of DLs

- How can time be modelled?
Temporal extensions of DLs

- How can time be modelled?
  - Point-based notion of time
  - Interval-based notion of time
Temporal extensions of DLs

- How can time be modelled?
  - Point-based notion of time
  - Interval-based notion of time

- How can the temporal dimension be handled?
Temporal extensions of DLs

- How can time be modelled?
  - Point-based notion of time
  - Interval-based notion of time

- How can the temporal dimension be handled?
  - Implicit notion of time: sequences of events through state-change representations
  - Explicit notion of time: definition of temporal operators and new formulae
Temporal extensions of DLs

- How can time be modelled?
  - Point-based notion of time
  - Interval-based notion of time

- How can the temporal dimension be handled?
  - Implicit notion of time: sequences of events through state-change representations
  - Explicit notion of time: definition of temporal operators and new formulae
  - Internal point of view: different states of an individual are modelled as different individual components
  - External point of view: an individual has different states in different moments
An Example

- How to become a doctor:
  - Be a PhD student for some years (5-6)
  - Defend a thesis
  - Become a doctor

Let’s try to model it with different temporal Description Logics!
Combining Description Logics and Tense Operators: ALCT

- **Combination of ALC with point-based modal temporal connectives**
  \[ \Diamond p, \Box p, \Diamond p, \mathcal{U} p, \mathcal{U} p \]

- **Time part of the semantic structure**
  \[ A^\mathcal{T} \subseteq \mathcal{T} \times \Delta^\mathcal{T} \]
  \[ R^\mathcal{T} \subseteq \mathcal{T} \times \Delta^\mathcal{T} \times \Delta^\mathcal{T} \]

**Recall:** “an individual has different states at different moments”

- **Temporal connectives can be applied only to concepts**
  \[ \Diamond C^\mathcal{T}_t := \{ a \in \Delta^\mathcal{T} | \exists t'. t' \leq t \land a \in C^\mathcal{T}_{t'} \} \]
An Example

- Every doctor has been a PhD student in the past

\[ \text{Doctor} \sqsubseteq \Diamond_p \text{PhD-Student} \]

- Every thesis defender is a PhD student that has been a PhD student in the past

\[ \text{Thesis-Defender} \sqsubseteq \text{PhD-Student} \cap \Diamond_p \text{PhD-Student} \]
An Example

- Every doctor has been a PhD student in the past

\[
\text{Doctor} \sqsubseteq \Diamond_p \text{PhD-Student}
\]

- Every thesis defender is a PhD student that has been a PhD student in the past

\[
\text{Thesis-Defender} \sqsubseteq \text{PhD-Student} \sqcap \Diamond_p \text{PhD-Student}
\]
Example

- Every doctor has been a PhD student in the past

\[
\text{Doctor} \sqsubseteq \Diamond_p \text{PhD-Student}
\]

- Every thesis defender is a PhD student that has been a PhD student in the past

\[
\text{Thesis-Defender} \sqsubseteq \text{PhD-Student} \sqcap \Diamond_p \text{PhD-Student}
\]
Interval-based Temporal DLs

- Schmiedel’s Formalism
  - Idea
  - Examples

- Bettini, The Undecidable Realm
  - Idea
  - Examples

- Artale and Franconi, Towards Decidable Logic
  - Idea
  - Examples
Interval-based Temporal DLs: Characteristics

- Interval-based

- Explicit, eg: alltime, ◊ TE

- Follows the external approach, eg: C<i, a> for temporal concept assertions and R<i, a, b> for temporal role assertions
Schmiedel’s Formalism: Idea

- The first of such an interval-based temporal DL (proposed in 1990)
- Underlying DL = $FLENR^-$ (no $T$, $\bot$, $\neg$, $\sqcup$ but with cardinality restrictions on roles, and role conjunction)
  - Temporal operators = at, alltime, sometime
  - Subsumption is argued to be undecidable
Schmiedel’s Formalism: Examples

**Concept:** PhD students during 1993
(at 1993 PhDStudent)

**Terminological Axiom:** Doctors were PhD students

\[ \text{Doctor} \models (\text{sometime}(x)(\text{metBy} \text{ now } x).(\text{at } x (\text{PhDStudent}))) \]
The Undecidable Realm: Idea

- Proposed by Bettini
  - Variable-free extension with existential and universal temporal quantification
    - Arbitrary relationships between more than two intervals can’t be represented
  - Satisfiability and subsumption are undecidable
  - Starting from the DL ALCN
  - Two concept constructors: ◊ TE.C and □ TE.C
The Undecidable Realm: Examples

Concept: Persons who become a doctor sometime
\textit{◊ after.\textit{Doctor}},
eg: \langle 1990, michael \rangle \text{belongs to this, if } \langle 1992, michael \rangle \text{belongs to Doctor}

Terminological Axiom: Doctors were PhD students
Doctor \sqsubseteq ◊ (\text{metBy}.\text{PhDStudent})
Towards Decidable Logics: Idea

- Developed by Artale and Franconi
  - Decidable: Expressivity is reduced, universal quantification on temporal variables has been eliminated
  - Underlying DL (most expressive): $T L$-ALCF
  - Temporal variables are introduced by the temporal existential quantifier $\Diamond$
    - excluding the predefined temporal var $\#$

Sami Al-Dhaheri
A Survey of Temporal Knowledge Representations
Towards Decidable Logics: Examples

Terminological Axiom: Doctors were PhD students

\[ \text{Doctor} \sqsubseteq \Diamond(x)(\# \text{metBy } x).\text{PhDStudent@x} \]

Terminological Axiom: PhD thesis defenders finish their PhDs

\[ \text{PhDThesisDefender} \sqsubseteq \Diamond(x)(\# \text{finishes } x).\text{PhDStudent@x} \]
The Semantic Web

“The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.”

[Berners-Lee et al, 2001]
The Semantic Web

Web pages are written in HTML

HTML describes the structure of information

HTML describes the syntax not the semantics

If computers machines can understand the meaning behind the information on the web, not just simply display it.

They can learn what we are interested in.

They can help us better find what we want.

This is really what the semantic web is all about.
The Semantic Web

• The goal of the semantic web is that it would be possible for machines to understand the information on the web rather than simply display.

• The major obstacle to this goal is the fact that most information on the web is designed solely for human consumption.

• This information should be structured in a way that machines can understand and process that information.

• The key technological threads that are currently employed in the development of Semantic Web are: eXtensible Markup Language (XML), Resource Description Framework (RDF), and Web Ontology Language (OWL)
RDF Triple

- RDF is a W3C standard for describing resources in the web.
- An RDF Triple \((s, p, o)\) is such that:
  - "s", "p" are URI-s, i.e., resources on the Web; "o" is a URI or a literal.
  - "s", "p", and "o" stand for "subject", "property", and "object".
- **RDF** is a general model for such triples with machine readable formats like RDF/XML.
RDF Graph Representation

http://www.w3.org/2000/10/swap/pim/contact#Person

http://www.w3.org/1999/02/22-rdf-syntax-ns#type

http://www.w3.org/People/EM/contact#me

http://www.w3.org/2000/10/swap/pim/contact#fullName

Eric Miller

http://www.w3.org/2000/10/swap/pim/contact#mailbox

mailto:em@w3.org

http://www.w3.org/2000/10/swap/pim/contact#personalTitle

Dr.
RDF Schema: Basic Ideas

• RDF is a universal language that enables users to describe their own vocabularies.
• But, RDF does not make assumption about any particular domain.
• It is up to user to define this in RDF schema.
What does RDF Schema add?

- Defines vocabulary for RDF
- Organizes this vocabulary in a typed hierarchy
  - Class, subClassOf, type
  - Property, subPropertyOf
  - domain, range
Temporal RDF
OWL

- Web Ontology Language (OWL) extends RDF by adding several other constructs such as `owl:class` (in addition to the `rdfs:class`), relationships between class and individuals, and property characteristics

- Follows Description Logic in having class, property, and individuals

- Allows setting the cardinality constraints
Reification

- \texttt{WorksFor(Employee, Company, TimeInterval)}
- representing the fact that an employee works for a company during a specific time interval.

\textbf{Figure 4: Reification}
4D fluent

- WorksFor(Employee, Company, TimeInterval)
- representing the fact that an employee works for a company during a specific time interval.

Figure 5: 4D fluent
Thank You