Querying with Łukasiewicz logic

Pietro Codara

Dipartimento di Informatica, Università degli Studi di Milano

(Joint work with S. Aguzzoli, T. Flaminio, B. Gerla, and D. Valota)

Overview

The Idea

*Test* an intended semantics for Łukasiewicz logic in a real-world situation.
Overview

The Idea

*Test* an intended semantics for Łukasiewicz logic in a real-world situation.

The Application

An automotive data-set. *We query the database via the pure language of the logic, i.e., via Łukasiewicz formulæ.*
Overview

The Idea

*Test* an intended semantics for Łukasiewicz logic in a real-world situation.

The Application

An automotive data-set. We query the database via the pure language of the logic, *i.e.*, via Łukasiewicz formulæ.

The Question

What can we ask to our database?
Łukasiewicz logic in brief (1)

Łukasiewicz logic can be semantically defined as a many-valued logic, as follows.
Łukasiewicz logic in brief (1)

Łukasiewicz logic can be semantically defined as a many-valued logic, as follows.
Let $\text{FORM}$ be the set of formulæ over propositional variables $X_1, X_2, \ldots$ in the language $\rightarrow, \neg, \bot$. 
Łukasiewicz logic in brief (1)

Łukasiewicz logic can be semantically defined as a many-valued logic, as follows.
Let FORM be the set of formulæ over propositional variables $X_1, X_2, \ldots$ in the language $\rightarrow, \neg, \bot$.
An assignment is a function $\mu: \text{FORM} \rightarrow [0, 1] \subseteq \mathbb{R}$ with values in the real unit interval such that, for any two $\alpha, \beta \in \text{FORM}$,
\[
\mu(\alpha \rightarrow \beta) = \min\{1, 1 - (\mu(\alpha) - \mu(\beta))\}
\]
\[
\mu(\neg \alpha) = 1 - \mu(\alpha)
\]
\[
\mu(\bot) = 0
\]
Łukasiewicz logic can be semantically defined as a many-valued logic, as follows.

Let $\textsc{form}$ be the set of formulæ over propositional variables $X_1, X_2, \ldots$ in the language $\to, \neg, \bot$.

An **assignment** is a function $\mu: \textsc{form} \to [0, 1] \subseteq \mathbb{R}$ with values in the real unit interval such that, for any two $\alpha, \beta \in \textsc{form}$,

\[
\begin{align*}
\mu(\alpha \to \beta) &= \min\{1, 1 - (\mu(\alpha) - \mu(\beta))\} \\
\mu(\neg \alpha) &= 1 - \mu(\alpha) \\
\mu(\bot) &= 0
\end{align*}
\]

A **tautology** is a formula $\alpha$ such that $\mu(\alpha) = 1$ for every assignment $\mu$. 
Łukasiewicz logic in brief (2)

Derived connectives \( \top, \lor, \land, \leftrightarrow, \oplus, \odot, \ominus \) are defined in the following table, for every formula \( \alpha \) and \( \beta \):

<table>
<thead>
<tr>
<th>Derived connective</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \top )</td>
<td>( \neg \bot )</td>
</tr>
<tr>
<td>( \alpha \lor \beta )</td>
<td>( \alpha \rightarrow \beta \rightarrow \beta )</td>
</tr>
<tr>
<td>( \alpha \land \beta )</td>
<td>( \neg(\neg \alpha \lor \neg \beta) )</td>
</tr>
<tr>
<td>( \alpha \leftrightarrow \beta )</td>
<td>( \alpha \rightarrow \beta \land \beta \rightarrow \alpha )</td>
</tr>
<tr>
<td>( \alpha \oplus \beta )</td>
<td>( \neg \alpha \rightarrow \beta )</td>
</tr>
<tr>
<td>( \alpha \odot \beta )</td>
<td>( \neg(\neg \alpha \oplus \neg \beta) )</td>
</tr>
<tr>
<td>( \alpha \ominus \beta )</td>
<td>( \neg(\alpha \rightarrow \beta) )</td>
</tr>
</tbody>
</table>

**Table**: Derived connectives in Łukasiewicz logic.

For each integer \( k > 0 \) and each formula \( \varphi \) let \( 1\varphi = \varphi \) and \((k + 1)\varphi = \varphi \oplus k\varphi \). Analogously, let \( \varphi^1 = \varphi \) and \( \varphi^{k+1} = \varphi \odot \varphi^k \).
Recent results show that Łukasiewicz logic is the logic suitable for dealing with a certain kind of vagueness – essentially the logic of "the more", "the less", and "the much more than".
Recent results show that Łukasiewicz logic is the logic suitable for dealing with a certain kind of vagueness – essentially the logic of "the more", "the less", and "the much more than".

Łukasiewicz logic can deal with properties which has a natural opposite. For instance, fast has as its opposite slow, with their obvious general meanings, while red, the property of being red, might not have a natural opposite.
Consider the propositions $P : "The car is fast"$, and $Q : "The car is cheap"$. Then,
Consider the propositions $P$: "The car is fast", and $Q$: "The car is cheap". Then,

1. $\neg P$ means "The car is slow";
The logic of vague proposition (2)

Consider the propositions $P : "The car is \text{ fast}"$, and $Q : "The car is \text{ cheap}"$. Then,

1. $\neg P$ means "The car is \text{ slow}";
2. $P \land Q$ means "The car is fast \text{ and} cheap";
Consider the propositions $P : "The car is fast"$, and $Q : "The car is cheap"$. Then,

1. $\neg P$ means "The car is slow";
2. $P \land Q$ means "The car is fast and cheap";
3. $P \lor Q$ means "The car is fast or cheap";
Consider the propositions $P$ : "The car is fast", and $Q$ : "The car is cheap". Then,

1. $\neg P$ means "The car is slow";
2. $P \land Q$ means "The car is fast and cheap";
3. $P \lor Q$ means "The car is fast or cheap";
4. $P \circ P$ means "The car is very fast", $P \circ P \circ P = P^3$ means "The car is very very fast", . . . ;
Consider the propositions $P : "The car is fast"$, and $Q : "The car is cheap"$. Then,

1. $\neg P$ means "The car is slow";
2. $P \land Q$ means "The car is fast and cheap";
3. $P \lor Q$ means "The car is fast or cheap";
4. $P \odot P$ means "The car is very fast", $P \odot P \odot P = P^3$ means "The car is very very fast", \ldots;
5. $P \oplus P$ means "The car is somewhat fast", \ldots;
The logic of vague proposition (2)

Consider the propositions $P$ : "The car is fast", and $Q$ : "The car is cheap". Then,

1. $\neg P$ means "The car is slow";
2. $P \land Q$ means "The car is fast and cheap";
3. $P \lor Q$ means "The car is fast or cheap";
4. $P \odot P$ means "The car is very fast", $P \odot P \odot P = P^3$ means "The car is very very fast", . . . ;
5. $P \oplus P$ means "The car is somewhat fast", . . . ;
6. $P \ominus Q$ means "The car is much more fast than cheap".
The database: data

Our model is a single database table where we have collected cars data. The table contains 4684 records, representing (a subset of) the cars on the Italian market in the year 2014.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Associated Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int(10) unsigned</td>
<td>–</td>
</tr>
<tr>
<td>manufacturer</td>
<td>varchar(50)</td>
<td>–</td>
</tr>
<tr>
<td>model</td>
<td>varchar(50)</td>
<td>–</td>
</tr>
<tr>
<td>trim</td>
<td>varchar(200)</td>
<td>–</td>
</tr>
<tr>
<td>price</td>
<td>int(11)</td>
<td>X0</td>
</tr>
<tr>
<td>length</td>
<td>int(11)</td>
<td>X1</td>
</tr>
<tr>
<td>width</td>
<td>int(11)</td>
<td>X2</td>
</tr>
<tr>
<td>height</td>
<td>int(11)</td>
<td>X3</td>
</tr>
<tr>
<td>fuel tank</td>
<td>int(11)</td>
<td>X4</td>
</tr>
<tr>
<td>seating capacity</td>
<td>tinyint(4)</td>
<td>X5</td>
</tr>
<tr>
<td>car segment</td>
<td>varchar(50)</td>
<td>–</td>
</tr>
<tr>
<td>drive</td>
<td>varchar(50)</td>
<td>–</td>
</tr>
<tr>
<td>fuel</td>
<td>varchar(50)</td>
<td>–</td>
</tr>
<tr>
<td>cubic capacity - cc</td>
<td>int(11)</td>
<td>X6</td>
</tr>
<tr>
<td>horsepower</td>
<td>int(11)</td>
<td>X7</td>
</tr>
<tr>
<td>power</td>
<td>int(11)</td>
<td>X8</td>
</tr>
<tr>
<td>environmental classification</td>
<td>varchar(10)</td>
<td>–</td>
</tr>
<tr>
<td>co2 emission</td>
<td>int(11)</td>
<td>X9</td>
</tr>
<tr>
<td>gearbox</td>
<td>varchar(50)</td>
<td>–</td>
</tr>
<tr>
<td>max speed</td>
<td>smallint(6)</td>
<td>X10</td>
</tr>
<tr>
<td>acceleration 0/100</td>
<td>decimal(5,2)</td>
<td>X11</td>
</tr>
<tr>
<td>urban cycle consumption</td>
<td>decimal(5,2)</td>
<td>X12</td>
</tr>
<tr>
<td>extra-urban cycle consumption</td>
<td>decimal(5,2)</td>
<td>X13</td>
</tr>
<tr>
<td>combined cycle consumption</td>
<td>decimal(5,2)</td>
<td>X14</td>
</tr>
</tbody>
</table>
The database: interface

Figure: The web page to submit queries.
Using linguistic hedges: an example

Our starting point is the expression of our desiderata in natural language: *I want a car with remarkable acceleration, but with reduced urban fuel consumption.*
Using linguistic hedges: an example

Our starting point is the expression of our **desiderata** in natural language: *I want a car with remarkable acceleration, but with reduced urban fuel consumption.*

The more natural way to query our database is thus to ask

\[ X_{11}^2 \land \neg X_{12} \]
Using linguistic hedges: an example

Our starting point is the expression of our desiderata in natural language: *I want a car with remarkable acceleration, but with reduced urban fuel consumption.*

The more natural way to query our database is thus to ask

\[ X_{11}^2 \land \neg X_{12} \]

The first 10 records of the answer set are:

- 4272  BMW Serie 3 GT 335d xDrive 2014 [0.793]
- 4275  BMW Serie 3 GT 335d xDrive 2014 [0.793]
- 2969  Audi SQ5 (8R) 3.0 TDI DPF quattro 2012 [0.763]
- 4287  BMW Serie 6 Coupe 640d xDrive 2012 [0.748]
- 4369  BMW Serie 6 GC 640d xDrive 2013 [0.748]
- 4433  BMW X4 xDrive 3.5d 2014 [0.748]
- 4462  BMW Serie 4 Cabrio 435d xDrive 2014 [0.748]
- 4468  BMW Serie 4 Cabrio 435d xDrive 2014 [0.748]
- 665   Infiniti Q50 S Hybrid 2013 [0.737]
- 3028  Audi A7 Sportback 3.0 TDI quattro 2012 [0.733]
Using linguistic hedges: updating queries

Suppose we do not want to spend so much money for our car, but we are looking, instead, for something very, very, (very) cheap.
Using linguistic hedges: updating queries

Suppose we do not want to spend so much money for our car, but we are looking, instead, for something very, very, (very) cheap.

We add a new clause talking about the cost of the car:

\[ X_{11}^2 \land \neg X_{12} \land \neg X_0^3 \]
Using linguistic hedges: updating queries

Suppose we do not want to spend so much money for our car, but we are looking, instead, for something very, very, (very) cheap.

We add a new clause talking about the cost of the car:

\[ X_{11}^2 \land \neg X_{12} \land \neg X_0^3 \]

The answer set seems to satisfy our request:

<table>
<thead>
<tr>
<th>Car Description</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ford Focus ST EcoBoost 250Cv 2012</td>
<td>4688</td>
</tr>
<tr>
<td>Audi S1 2.0 TFSI quattro 2014</td>
<td>3003</td>
</tr>
<tr>
<td>Volkswagen Golf VII 2.0 TSI GTI 2013</td>
<td>2745</td>
</tr>
<tr>
<td>Renault Megane III Coupe 2.0 TCIe 265 2014</td>
<td>2205</td>
</tr>
<tr>
<td>Ford Focus Wagon ST 2012</td>
<td>4781</td>
</tr>
<tr>
<td>Audi S1 Sportback 2.0 TFSI quattro 2014</td>
<td>3000</td>
</tr>
<tr>
<td>Seat Leon SC TSI Cupra 2014</td>
<td>478</td>
</tr>
<tr>
<td>Opel Astra J GTC Turbo OPC 2012</td>
<td>3292</td>
</tr>
<tr>
<td>Seat Leon 2.0 TSI Cupra 2014</td>
<td>496</td>
</tr>
<tr>
<td>Volkswagen Golf VII 2.0 TSI GTI 2013</td>
<td>2736</td>
</tr>
<tr>
<td>Renault Clio IV 1.6 TCIe 200 Monaco GP 2014</td>
<td>52</td>
</tr>
</tbody>
</table>
Using linguistic hedges: comments

- One of the most interesting features of Łukasiewicz querying is the capability of using linguistic hedges such as somewhat or very.
Using linguistic hedges: comments

- One of the most interesting features of Łukasiewicz querying is the capability of using linguistic hedges such as *somewhat* or *very*.
- Queries written as formulæ in Ł are easily updatable.
One of the most interesting features of Łukasiewicz querying is the capability of using linguistic hedges such as somewhat or very.

Queries written as formulæ in $Ł$ are easily updatable.

When we conjunct new conditions, the truth values of the top records of the answer set is decreasing. We shall overcome this obstacle by asking for our final query to be somewhat satisfied. For instance, we can rewrite $X_{11}^2 \land \neg X_{12} \land \neg X_{03}$ as $2(X_{11}^2 \land \neg X_{12} \land \neg X_{03})$. 

It is always possible to rewrite a traditional query in the form $X \leq k$ or $X \geq k$, for $k \in [0, 1]$ a rational value, in a Łukasiewicz query which provide exactly the same result. Nevertheless, this procedure produces very artificial values for exponents and multipliers. For example, $X_{11} \geq 0.875$, and $X_{12} \leq 0.25$ becomes $20(X_{11}^8 \land \neg X_{412} \land \neg X_{03})$. 

Using linguistic hedges: comments
Using linguistic hedges: comments

- One of the most interesting features of Łukasiewicz querying is the capability of using linguistic hedges such as somewhat or very.
- Queries written as formulæ in Ł are easily updatable.
- When we conjunct new conditions, the truth values of the top records of the answer set is decreasing. We shall overcome this obstacle by asking for our final query to be somewhat satisfied. For instance, we can rewrite
  \[ X_{11}^2 \land \neg X_{12} \land \neg X_0^3 \]  as \[ 2(X_{11}^2 \land \neg X_{12} \land \neg X_0^3). \]
- It is always possible to rewrite a traditional query in the form \( X \leq k \) or \( X \geq k \), for \( k \in [0, 1] \) a rational value, in a Łukasiewicz query which provide exactly the same result. Nevertheless, this procedure produces very artificial values for exponents and multipliers. For example, \( X_{11} \geq 0.875 \), and \( X_{12} \leq 0.25 \) becomes \[ 20(X_{11}^8 \land \neg X_{12}^4). \]
The minus connective ⊖

Suppose you can describe, via a formula $\alpha$, what (in your opinion) makes a car a good car. Further, suppose you can describe, via a formula $\beta$, what makes a car a bad car.
The minus connective $\ominus$

- Suppose you can describe, via a formula $\alpha$, what (in your opinion) makes a car a good car. Further, suppose you can describe, via a formula $\beta$, what makes a car a bad car.
- Consider the formula $\gamma = \alpha \land \neg \beta$. Intuitively, $\gamma$ describe your perfect car, with all pros and no cons. Let $A$ be a car satisfying $\alpha$ in degree 0.5, and $\beta$ in degree 0.5. Thus, $A$ satisfy $\gamma$ in degree 0.5.
The minus connective

- Suppose you can describe, via a formula $\alpha$, what (in your opinion) makes a car a good car. Further, suppose you can describe, via a formula $\beta$, what makes a car a bad car.
- Consider the formula $\gamma = \alpha \land \neg \beta$. Intuitively, $\gamma$ describes your perfect car, with all pros and no cons. Let $A$ be a car satisfying $\alpha$ in degree 0.5, and $\beta$ in degree 0.5. Thus, $A$ satisfy $\gamma$ in degree 0.5.
- Let now $\phi = \alpha - \beta$. In natural language, $\phi$ asks for a car which is much more $\alpha$ than $\beta$. The truth value of $\phi$ when evaluated in $A$ is 0. While according to $\gamma$, the car $A$ is a medium car, $\phi$ discard the same car.
Suppose you can describe, via a formula $\alpha$, what (in your opinion) makes a car a good car. Further, suppose you can describe, via a formula $\beta$, what makes a car a bad car.

Consider the formula $\gamma = \alpha \land \neg \beta$. Intuitively, $\gamma$ describes your perfect car, with all pros and no cons. Let $A$ be a car satisfying $\alpha$ in degree 0.5, and $\beta$ in degree 0.5. Thus, $A$ satisfy $\gamma$ in degree 0.5.

Let now $\varphi = \alpha - \beta$. In natural language, $\varphi$ asks for a car which is much more $\alpha$ than $\beta$. The truth value of $\varphi$ when evaluated in $A$ is 0. While according to $\gamma$, the car $A$ is a medium car, $\varphi$ discard the same car.

An assertion like $\varphi$ can be interpreted as a measure of the level of satisfaction of the buyer: satisfaction begin when pros overcome cons, and is maximal when we have all possible pros, and no cons.
The minus connective $\ominus$: an example

The buyer, asked for a definition of a *good car*: "A *good car* has a *remarkable* acceleration, but a *reduced* urban fuel consumption". In symbols: $X_{11}^2 \land \neg X_{12}$. 
The minus connective $\ominus$: an example

- The buyer, asked for a definition of a *good car*: "A *good car* has a *remarkable* acceleration, but a *reduced* urban fuel consumption". In symbols: $X^2_{11} \land \neg X_{12}$.

- The buyer, asked for a definition of a *bad car*: "A *bad car* is an expensive car". In symbols: $X_0$. 
The minus connective $\ominus$: an example

- The buyer, asked for a definition of a good car: "A good car has a remarkable acceleration, but a reduced urban fuel consumption". In symbols: $X^2_{11} \land \neg X_{12}$.

- The buyer, asked for a definition of a bad car: "A bad car is an expensive car". In symbols: $X_0$.

- In the preceding example, we have forced a car to be cheap, using $\land$. We want, instead, to assert that we do not make any difference between a good car with a medium price, and a medium car with a low price. We are asking for a car that is much more good than bad.
The minus connective $\ominus$: an example, cont.

To make such request, we use the following query.

$$X_{11}^2 \land \neg X_{12} \ominus \neg X_3^0$$
The minus connective $\ominus$: an example, cont.

To make such request, we use the following query.

$$X_{11}^2 \land \neg X_{12} \ominus \neg X_0^3$$

The answer we obtain querying the database is the following.

<table>
<thead>
<tr>
<th>ID</th>
<th>Model Name</th>
<th>Year</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>3003</td>
<td>Audi S1 2.0 TFSI quattro 2014</td>
<td></td>
<td>0.510</td>
</tr>
<tr>
<td>3000</td>
<td>Audi S1 Sportback 2.0 TFSI quattro 2014</td>
<td></td>
<td>0.490</td>
</tr>
<tr>
<td>478</td>
<td>Seat Leon SC 2.0 TSI Cupra 2014</td>
<td></td>
<td>0.490</td>
</tr>
<tr>
<td>496</td>
<td>Seat Leon 2.0 TSI Cupra 2014</td>
<td></td>
<td>0.488</td>
</tr>
<tr>
<td>2988</td>
<td>Audi S3 2.0 TFSI quattro 2013</td>
<td></td>
<td>0.481</td>
</tr>
<tr>
<td>487</td>
<td>Seat Leon SC 2.0 TSI Cupra 2014</td>
<td></td>
<td>0.480</td>
</tr>
<tr>
<td>4275</td>
<td>BMW Serie 3 GT 335d xDrive 2014</td>
<td></td>
<td>0.480</td>
</tr>
<tr>
<td>497</td>
<td>Seat Leon 2.0 TSI Cupra 2014</td>
<td></td>
<td>0.478</td>
</tr>
<tr>
<td>2993</td>
<td>Audi S3 Sportback 2.0 TFSI quattro 2013</td>
<td></td>
<td>0.476</td>
</tr>
<tr>
<td>1595</td>
<td>BMW Serie 2 Coupe 228i 2014</td>
<td></td>
<td>0.472</td>
</tr>
</tbody>
</table>
The pure Łukasiewicz logic can be effectively, and efficiently, used to treat real problems: in our case, to query an online database of cars.
Conclusion, and further work

- The pure Łukasiewicz logic can be effectively, and efficiently, used to treat real problems: in our case, to query an online database of cars.

- Much work remains to be done on this front to develop an online system which is meant for a general user, typically not enough expert in logic to query with logical formulæ.
Conclusion, and further work

- The pure Łukasiewicz logic can be effectively, and efficiently, used to treat real problems: in our case, to query an online database of cars.
- Much work remains to be done on this front to develop an online system which is meant for a general user, typically not enough expert in logic to query with logical formulæ.
- To this end, we need to design the mechanisms of interaction between user and system, and
Conclusion, and further work

- The pure Łukasiewicz logic can be effectively, and efficiently, used to treat real problems: in our case, to query an online database of cars.

- Much work remains to be done on this front to develop an online system which is meant for a general user, typically not enough expert in logic to query with logical formulæ.

- To this end, we need to design the mechanisms of interaction between user and system, and

- ... the system must be equipped with an algorithm able to translate the user’s desiderata in a Łukasiewicz formula.
Thank you for your attention.