WOLED: A Tool for Online Learning Weighted Answer Set Rules for Temporal Reasoning Under Uncertainty

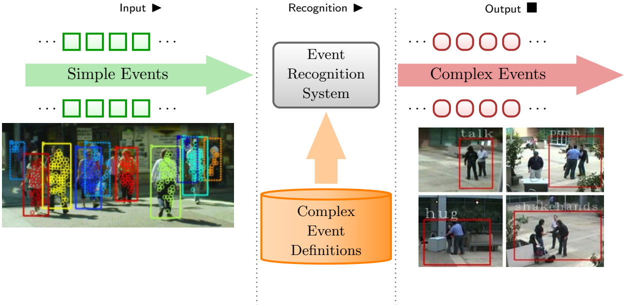
Nikos Katzouris[1], Alexander Artikis[2,1]

http://cer.iit.demokritos.gr

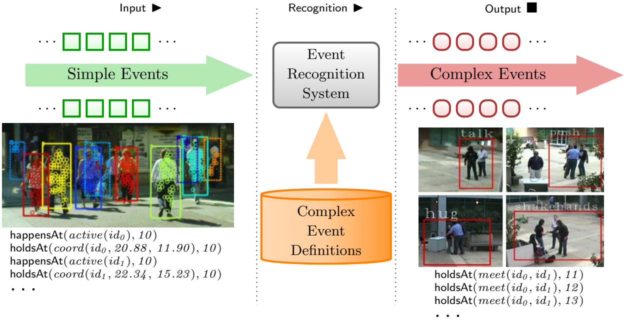
[1]National Center for Scientific Research Demokritos, Athens, Greece
[2]University of piraeus, Piraeus, Greece

KR 2020

Application: Complex Event Recognition



Application: Complex Event Recognition



Application: Complex Event Recognition

Event Calculus as a Reasoning Engine

holdsAt$(F, T+1) \leftarrow$
initiatedAt$F, T)$

holdsAtAt$(F, T+1) \leftarrow$
holdsAt(F, T),
not terminatedAt(F, T).

Very efficient inference: *Artikis et al. An Event Calculus for Event Recognition, TKDE, 2015.*

Input ▶ Recognition ▶ Output ■

Simple Events → Event Recognition System → Complex Events

Complex Event Definitions

happensAt($active(id_0)$, 10)
holdsAt($coord(id_0, 20.88, 11.90)$, 10)
happensAt($active(id_1)$, 10)
holdsAt($coord(id_1, 22.34, 15.23)$, 10)
. . .

holdsAt($meet(id_0, id_1)$, 11)
holdsAt($meet(id_0, id_1)$, 12)
holdsAt($meet(id_0, id_1)$, 13)
. . .

Application: Complex Event Recognition

Event Calculus as a Reasoning Engine

$\mathsf{holdsAt}(F,\ T+1) \leftarrow$
$\quad \mathsf{initiatedAt} F,\ T)$

$\mathsf{holdsAtAt}(F,\ T+1) \leftarrow$
$\quad \mathsf{holdsAt}(F,\ T),$
$\quad \mathit{not}\ \mathsf{terminatedAt}(F,\ T).$

Very efficient inference: *Artikis et al. An Event Calculus for Event Recognition, TKDE, 2015.*

Input ▶ Recognition ▶ Output ■

Simple Events → Event Recognition System → Complex Events

Complex Event Definitions

$\mathsf{happensAt}(\mathit{active}(id_0),\ 10)$
$\mathsf{holdsAt}(\mathit{coord}(id_0,\ 20.88,\ 11.90),\ 10)$
$\mathsf{happensAt}(\mathit{active}(id_1),\ 10)$
$\mathsf{holdsAt}(\mathit{coord}(id_1,\ 22.34,\ 15.23),\ 10)$
...

$\mathsf{initiatedAt}(\mathit{meet}(X,\ Y),\ T) \leftarrow$
$\quad \mathsf{happensAt}(\mathit{active}(X),\ T),$
$\quad \mathsf{happensAt}(\mathit{active}(Y),\ T),$
$\quad \mathsf{holdsAt}(\mathit{close}(X,\ Y,\ 25),\ T).$

$\mathsf{terminatedAt}(\mathit{meet}(X,\ Y),\ T) \leftarrow$
$\quad \mathsf{happensAt}(\mathit{walking}(X),\ T),$
$\quad \mathit{not}\ \mathsf{holdsAt}(\mathit{close}(X,\ Y,\ 25),\ T).$

$\mathsf{holdsAt}(\mathit{meet}(id_0,\ id_1),\ 11)$
$\mathsf{holdsAt}(\mathit{meet}(id_0,\ id_1),\ 12)$
$\mathsf{holdsAt}(\mathit{meet}(id_0,\ id_1),\ 13)$
...

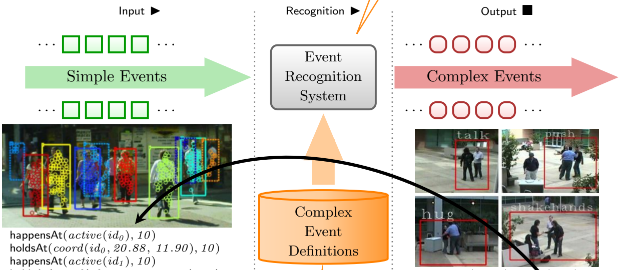
Learning for Complex Event Recognition

Event Calculus as a Reasoning Engine

$\mathsf{holdsAt}(F, T+1) \leftarrow$
$\quad \mathsf{initiatedAt}F, T)$

$\mathsf{holdsAtAt}(F, T+1) \leftarrow$
$\quad \mathsf{holdsAt}(F, T),$
$\quad \mathit{not}\ \mathsf{terminatedAt}(F, T).$

Very efficient inference: *Artikis et al. An Event Calculus for Event Recognition, TKDE, 2015.*



$\mathsf{happensAt}(active(id_0), 10)$
$\mathsf{holdsAt}(coord(id_0, 20.88, 11.90), 10)$
$\mathsf{happensAt}(active(id_1), 10)$
$\mathsf{holdsAt}(coord(id_1, 22.34, 15.23), 10)$
...

$\mathsf{holdsAt}(meet(id_0, id_1), 11)$
$\mathsf{holdsAt}(meet(id_0, id_1), 12)$
$\mathsf{holdsAt}(meet(id_0, id_1), 13)$
...

Learn this

$\mathsf{initiatedAt}(meet(X, Y), T) \leftarrow$
$\quad \mathsf{happensAt}(active(X), T),$
$\quad \mathsf{happensAt}(active(Y), T),$
$\quad \mathsf{holdsAt}(close(X, Y, 25), T).$

$\mathsf{terminatedAt}(meet(X, Y), T) \leftarrow$
$\quad \mathsf{happensAt}(walking(X), T),$
$\quad \mathit{not}\ \mathsf{holdsAt}(close(X, Y, 25), T).$

From These

Learning Requirements

- Event recognition applications deal with noisy data streams.
  - Resilience to noise & uncertainty:
    - Statistical Relational Learning.
    - Logical representations + probability.
  - Big data, data streams.
    - Online, single-pass learning.

Learning Requirements

- Event recognition applications deal with noisy data streams.
  - Resilience to noise & uncertainty:
    - Statistical Relational Learning.
    - Logical representations + probability.
  - Big data, data streams.
    - Online, single-pass learning.
- Statistical Relational Learning:
  - Rules' structure learning.
    - Inductive Logic Programming
  - Weight learning.
    - Gradient-based techniques.

Statistical Relational Learning in Answer Set Programming

- Why?
  - Non-monotonic semantics.
  - Sophisticated off-the-self ASP solvers.
  - Structure & weight learning tasks easily encoded as optimization problems in ASP.

Statistical Relational Learning in Answer Set Programming

- Why?
  - Non-monotonic semantics.
  - Sophisticated off-the-self ASP solvers.
  - Structure & weight learning tasks easily encoded as optimization problems in ASP.
- How?
  - Setting very similar to Markov Logic Networks.
  - Real-valued weights attached to rules in an ASP program Π.
  - Larger weights, larger confidence to rules
  - Weights define a prob. distribution over answer sets of Π.
  - Lee & Young, *Weighted rules under the stable model semantics*, KR 2016.

Probabilistic MAP Inference

- Task: find most probable answer set.
- Turns out to be a weighted MaxSat problem:
  - Find an answer set that maximizes the sum of weights of satisfied rules.

Probabilistic MAP Inference

- Task: find most probable answer set.
- Turns out to be a weighted MaxSat problem:
  - Find an answer set that maximizes the sum of weights of satisfied rules.
- Handled directly by an ASP solver:

$$\mathtt{head_i} \leftarrow \mathtt{satisfied(i)}$$
$$\{\mathtt{satisfied(i)}\} \leftarrow \mathtt{body_i}$$
$$:\sim \mathtt{satisfied(i)}.\ [-w_i]$$

$\mathtt{head_i} \leftarrow \mathtt{body_i}$ *is the i-th rule with weight* w_i.

Weight Learning

- Compare results in MAP-inferred state with true state.
- Update weights according to mistakes.

Weight Learning

- Compare results in MAP-inferred state with true state.
- Update weights according to mistakes.

- AdaGrad-based weight update rule:

Previous weight of the i-th rule

Learning rate

Rule's current mistakes

Regularization rate

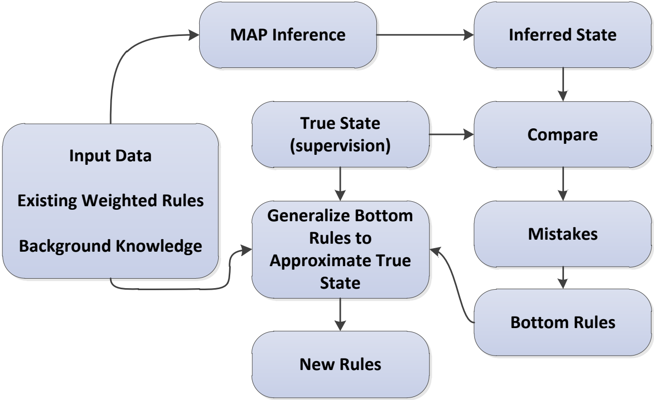
$$w_i^{t+1} = sign(w_i^t - \frac{\eta}{C_i^t} \Delta g_i^t)\ max\{0, |w_i^t - \frac{\eta}{C_i^t} \Delta g_i^t| - \lambda \frac{\eta}{C_i^t}\}$$

Current weight of the i-th rule

Term proportional to the rule's accumulated past mistakes

- Δg_i^t (i-th rule's mistakes at time t): difference in rule's true groundings in the true state and the MAP-inferred state.

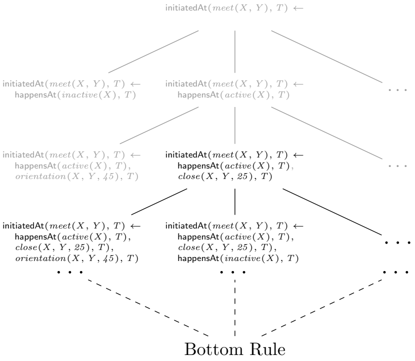
Structure Learning I: Learning New Rules from Mistakes



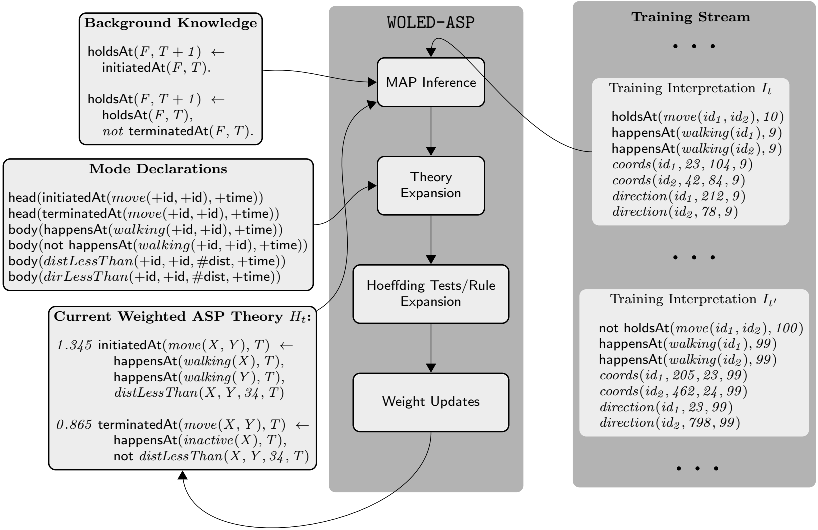
- Techniques from non-monotonic Inductive Logic Programming.
- Reasoning with existing weighted rules and generalizing new bottom rules part of the **same optimization process in ASP**.

Structure Learning I: Revising Existing rules

- As new data arrive the structure of rules often need to be revised.
  - Specialize rules.
- Online Hill-Climbing via Hoeffding tests.
- Using a small part of the input stream at each specialization decision point.



Putting it All Together

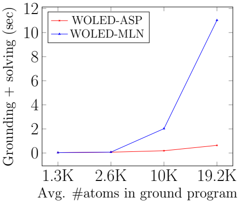


Experimental Evaluation

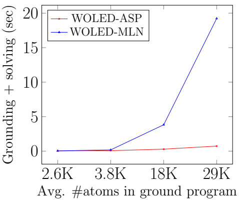
- Applications & datasets:
  - Activity recognition
    - 28 videos transcribed in logical form.
    - Target events: Two persons moving together, or meeting each other.
  - Maritime Surveillance
    - AIS signals of vessels sailing around the area of Brest, France.
    - 6 months worth of data.
    - Target event: Suspicious vessel Rendezvous.
  - Vehicle fleet management
    - Signals from on-vehicle sensors.
    - 1 month worth of data.
    - Target event: Dangerous driving.

Scalability of MAP Inference (MLN vs. ASP)

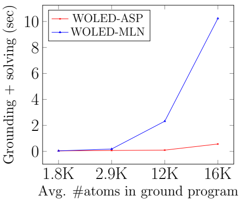
- Fixed event pattern set.
- Map inference for weight learning.
- Varying batch sizes.



Moving



Vessel Rendezvous



Dangerous Driving

- WOLED-ASP
  - Clingo
- WOLED-MLN
  - `LoMRF`[1] lib for Markov Logic.
  - `lpsolve`[2] lib for Integer Linear Programming.

[1] https://github.com/anskarl/LoMRF

[2] https://sourceforge.net/projects/lpsolve

Learning Performance

	Method	Prequential Loss	F_1-score (test set)	Theory size	Inference Time (sec)	Pred. Compl. Time (sec)	Total Time (sec)
Moving	WOLED-ASP	**1.723**	**0.821**	26	15	–	112
	WOLED-MLN	2.817	0.801	47	187	28	478
	OLED	3.755	0.730	**24**	**13**	–	74
	HandCrafted	6.342	0.637	28	–	–	–
	HandCrafted-WL	4.343	0.702	28	16	–	**52**
Meeting	WOLED-ASP	**1.212**	**0.887**	34	**12**	–	82
	WOLED-MLN	2.554	0.841	56	134	12	145
	OLED	3.224	0.782	42	10	–	36
	HandCrafted	5.734	0.735	**23**	–	–	–
	HandCrafted-WL	4.024	0.753	**23**	13	–	**31**
Rendezvous	WOLED-ASP	**0.023**	0.98	18	647	–	4,856
	WOLED-MLN	0.088	0.98	18	2,923	434	6,218
	OLED	0.092	0.98	18	**623**	–	**4,688**
Dang.Drive	WOLED-ASP	**0.045**	0.99	**21**	341	–	2,465
	WOLED-MLN	1.234	0.99	28	926	287	3,882
	OLED	1.756	0.99	**21**	**312**	–	**2,435**

Summary

- An online structure & weight learner entirely implemented in ASP.
- Significantly more efficient & simpler to use than MLN.
  - Single back-end tool – Clingo.
- Structure & weight learning tightly coupled.
- `https://github.com/nkatzz/ORL`

Future work:

- Concept drift.
- Distributed learning.