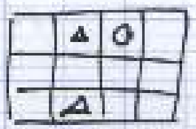
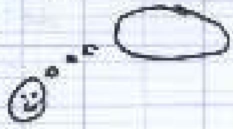
Hintikka's world.

Goals:

1) Promoting the tool HW
2) Epistemic logic in CS
   a) Knowledge-based programs.
   b) Asynchrony

1) Tool HW

a) History

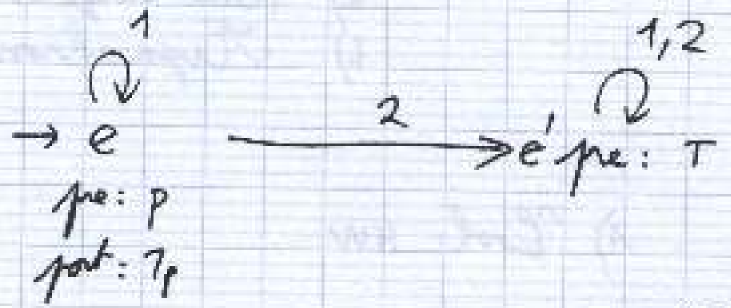
Tarski's world	FO	
Kripke's world	ML	tableau prover
Hintikka's world	DEL	

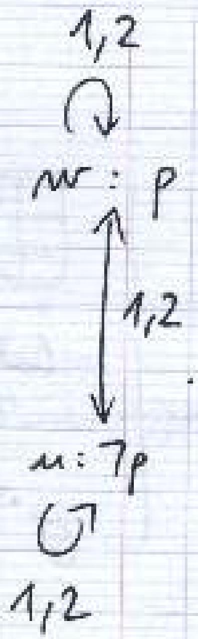
b) Practical session

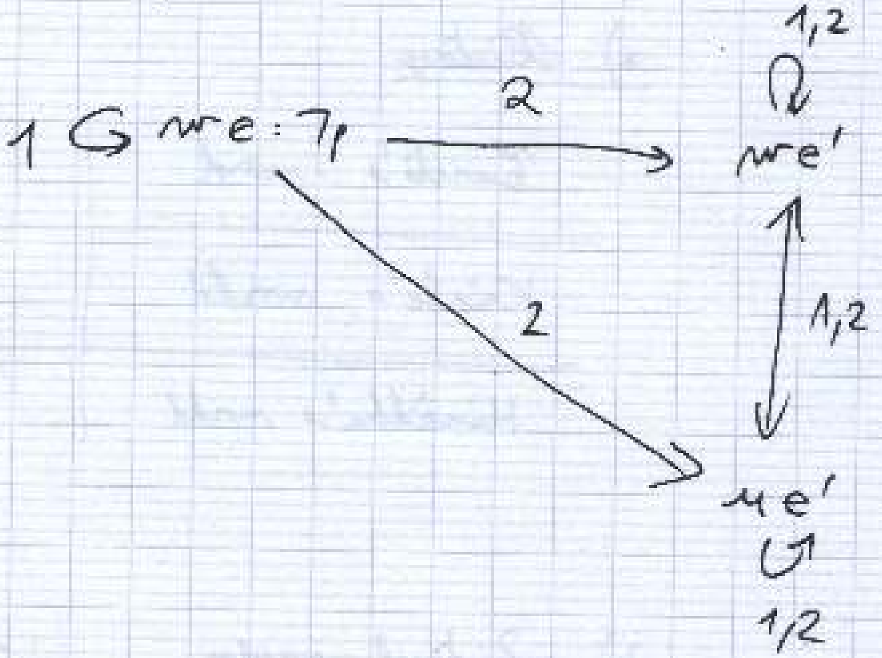
google Hintikka's world.

b) Dynamic epistemic logic (behind the scene)

epistemic situation	Kripke model
events	event model







d) Model checking

$\mathcal{M}, w \models \varphi$

without * e.g $\mathcal{M}, w \models K_a [\mathcal{E}, e][\mathcal{E}', e'] K_b p$

⇝ decidable

⇝ PSPACE-c even if models are symbolic

with * : e.g $\mathcal{M}, w \models [((\mathcal{E}, e) \cup (\mathcal{E}', e'))^*] K_b p$

⇝ undecidable.

2) Epistemic logic in CS

a) Knowledge-based programs

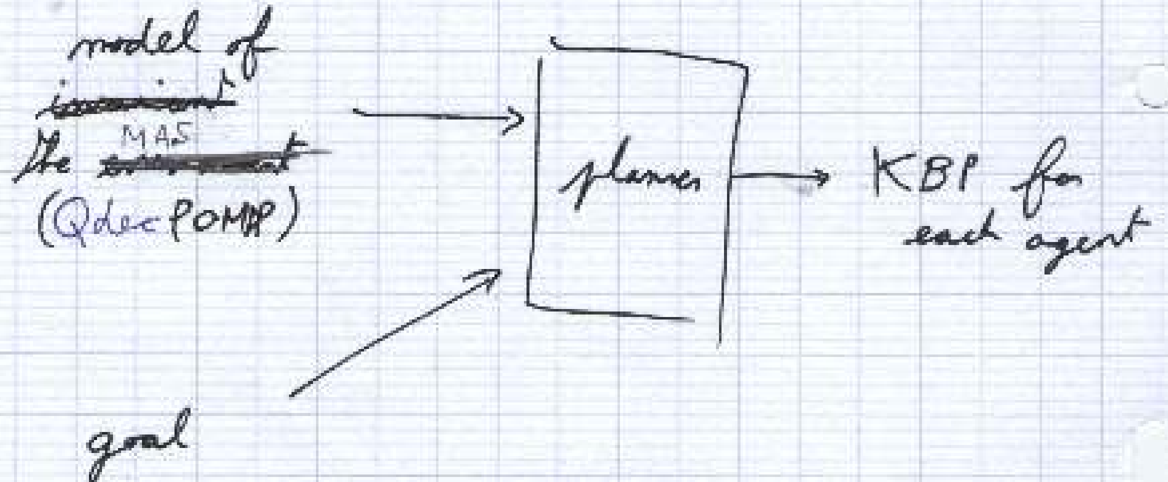
Ex:

```
A
B
if ( Ka p)
{      A
else
{      B

while ¬Ka q
{      C
```

→ easy to read by humans
→ explainable AI

Ex of use in the future:



[AAAI 2018]

Goal: define a <u>semantics</u> for KBPs.

QdecPOMDP
KBPs) $\rightsquigarrow$ Kripke's models.

$\rightsquigarrow$ demo

<u>Déf</u>: (QdecPOMDP) a non-det transition system Π with transitions of the form:

$$\text{state } s \xrightarrow{act_1,\ act_2,\ obs_1,\ obs_2} \text{state } s'$$

e.g. turn right / left ... $T^\circ > 0$ there is a wall ...

and with an ~~set~~ subset S° of initial states.

<u>Déf</u>: ($\mathcal{M}_0$) Given Π, $(\pi_a)_{a \in Ag}$, $\mathcal{M}_0$ is

Worlds of $\mathcal{M}_0$: $s^\circ c^\circ$ where $s^\circ \in S^\circ$
c° is a vector of initial program counters.

$\sim_a$: complete relations over worlds.

Déf: ($\mathcal{M}_{t+1}$ from $\mathcal{M}_t$, by induc. on t.)

Worlds of $\mathcal{M}_{t+1}$ are of the form:

history of length t

| world of $\mathcal{M}_t$ $s^t c^t$ | w^t s^{t+1} c^{t+1} |

- world of $\mathcal{M}_t$ $s^t c^t$: should satisfy the guard of c^{t+1}
- s^{t+1}: gives the valuation
- $w^t\, s^{t+1}$: nature's outcome of act in c^{t+1}
- c^{t+1}: succ of c^t.

$\boxed{h^t}\; w^t\, s^{t+1}\, c^{t+1}$

$\sim_{t+1}$

$\boxed{h'^t}\; w'^t\, s'^{t+1}\, c'^{t+1}$

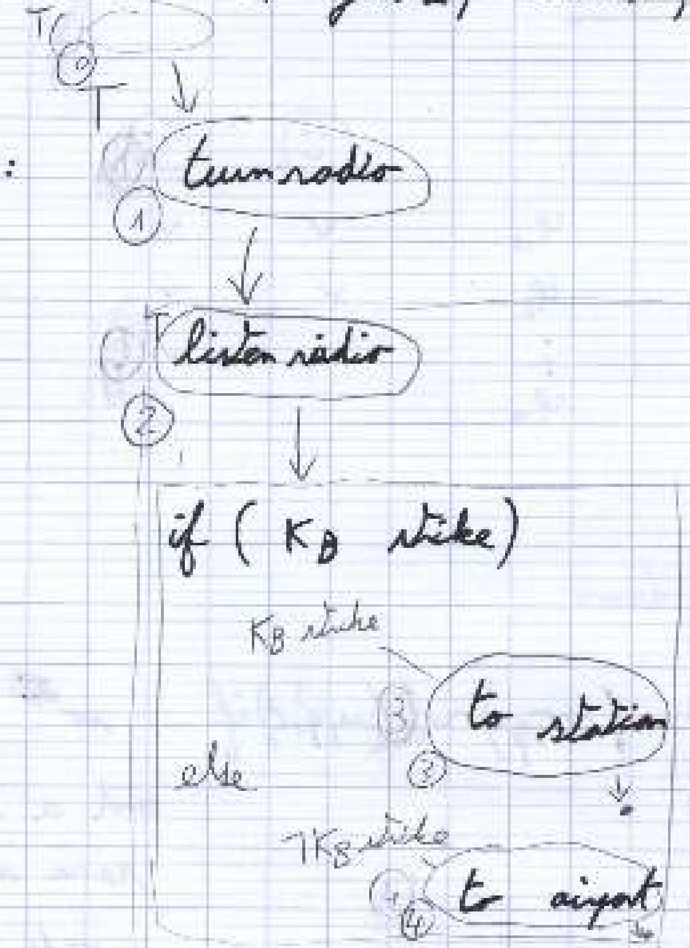
if 1) $h^t \sim_t h'^t$

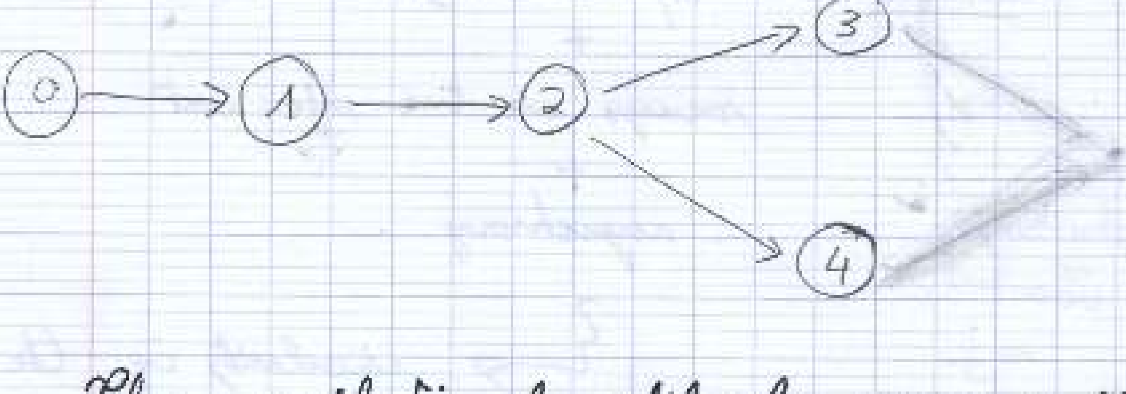
2) $w_i^t = w_i^{t'}$.

<u>Déf</u> : (program counter)

< guard, action, continuation >.

Ex:



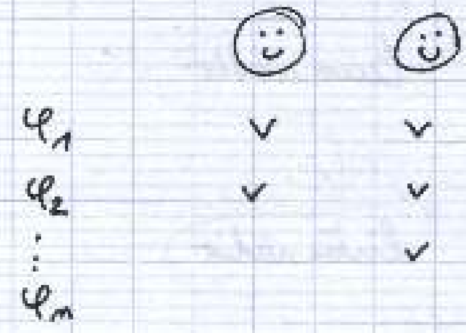


<u>Th</u>: verificat° of while-free programs: PSPACE-c

verificat° of while-programs: undecidable.

b) Asynchrony

Architecture :



⇝ demo

Idea : $(w,q,c) \sim_a (u,q',c')$ if $w \sim_a u$

and a has received the same msg and in the same order in w and u.

$c[i] = c'[i] \,\&\, q[1..c[i]] = q'[1,..c'[i]]$

Bad news : epistemic messages

+

messages are true when sent

+

asynchrony

⟶ circularity in the truth condition def.

[Sophie Knight, Maubert, —]

Ex:

$w, \sqsubset \models K_a p$?

$\wr^a$

u [$K_a p$ ⋮ ...]

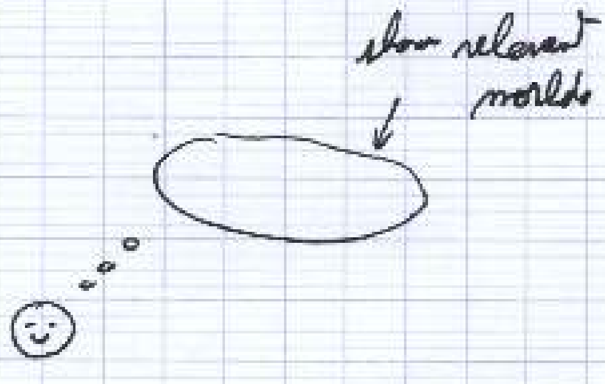
$u \sqsubset \models K_a p$?

Conclusion - Future work

Tools

1) Design

2) Symbolic models



3) Extensions

Research

1) Face undecidability of verificat.
   → bounded planning
   → approx.
2) Strategic reasoning = KBPs.
3) Face the circularity pb due to asynchrony.