Autonomous Decision-Making with Incomplete Information and Safety Rules based on Non-Monotonic Reasoning

José-Luis Vilchis-Medina, Charles Lesire-Cabaniols[a], Karen Godary-Dejean[b]

ONERA – The French Aerospace Lab[a]
LIRMM, Université de Montpellier, CNRS[b]

ENSTA Bretagne
November 16, 2023

Overview

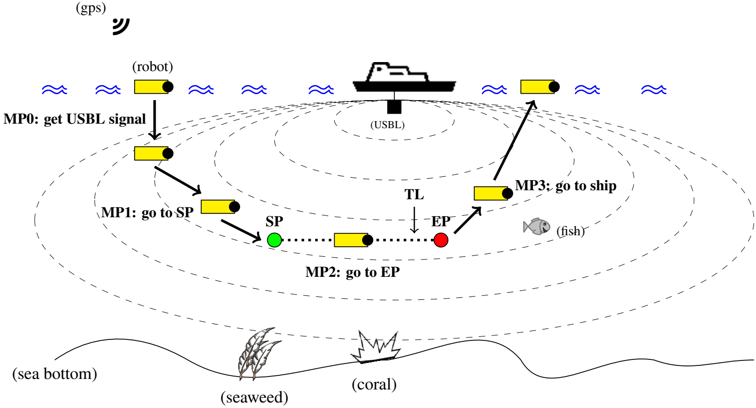
Description of the robot

Underwater robot

- dimension 93cm L, 41cm W, 23cm H,
- micro-computer Intel Atom Z8000 processor,
- 4 batteries, 8 motors, weight ≈30Kgs

Sensor name	ID	Protocol
Leak indicator	SOS-Leak-Sensor	Digital
Flasher	Led	Digital
Bar30	MS5837	i2c
IMU	BNO055	i2c
Camera	-	IP
GPS–Robot	DP0104	uart
Echo-sounder	Ping Sonar	uart
SeaTrac	X150 USBL Transponder	uart

Example of a normal mission : *"transect"*



Problematic

Generally, autonomous robots are confronted with unexpected situations due to multiples causes:

- changes in the environment,
- uncertain information,
- failures, etc.

Example (in our case):

temperature increase, water leakage, non geo-referenced location, exceeded depth, mission overrun time, drifting from transect line, etc.

Solution/Proposition

Goal reasoning: it is to decide on current information, while always considering the objective(s) and safety of the robot.

Overview

State of the art

- KnowRob framework[1]:
  - vague information treatment to perform tasks: "set the table", "clean up"...
  - Description Logic for knowledge representation,
  - central knowledge consultation via Prolog (classical proving theorems)
- Answer-Set Programming (ASP):
  - declarative programming and stable model paradigm[2],
  - generally has difficulties to reason on all classes of stable models[3],
  - does not support free variables and difficult to debug a program
- I proposed (on my Ph.D. thesis):
  - a formalization for modeling a solar glider's piloting behavior using a non-monotonic logic,
  - the use of Default Logic (DL) and Prolog (non-monotonic reasoner),
  - a framework for the study of resilient systems using DL.

[1] Tenorth, Moritz, and Michael Beetz. "KnowRob: A knowledge processing infrastructure for cognition-enabled robots." (2013).

[2] Erdem, Esra, Erdi Aker, and Volkan Patoglu. "Answer set programming for collaborative housekeeping robotics: representation, reasoning, and execution." Intelligent Service Robotics 5.4 (2012): 275-291.

[3] Elkhatib, Omar, Enrico Pontelli, and Tran Cao Son. "ASP − PROLOG: A System for Reasoning about Answer Set Programs in Prolog." International Symposium on Practical Aspects of Declarative Languages. Springer, Berlin, Heidelberg, 2004.

Overview

Non-monotonic Reasoning

The most important proposal by:

J. McCarthy (*Circumscription '80*), **R. Reiter** (*Default logic '80*)...

- New information can invalidate previous conclusions,
- Resolve contradictions,
- Reasoning about knowledge,
- Rational conclusions from partial information.

Non-monotonic Reasoning

The most important proposal by:

J. McCarthy (*Circumscription '80*), **R. Reiter** (*Default logic '80*)...

- New information can invalidate previous conclusions,
- Resolve contradictions,
- Reasoning about knowledge,
- Rational conclusions from partial information.

Formally, monotonicity:

$$A \rightarrow \omega$$
$$A \cup B \rightarrow \omega$$

Default Logic [Reiter]

Definition

A default theory is a pair $\Delta = (D, W)$, where D is a set of defaults and W is a set of formulas in FOL.

- A default d is: $\frac{A(X):B(X)}{C(X)}$
- $A(X), B(X), C(X)$ are well-formed formulas
- $X = (x_1, x_2, x_3, \ldots, x_n)$ is a vector of free variables(non-quantified).

Intuitively a default means, **"if $A(X)$ is true, and there is no evidence that $B(X)$ might be false, then $C(X)$ can be true"**.

Default Logic [Reiter]

Definition

A default theory is a pair $\Delta = (D, W)$, where D is a set of defaults and W is a set of formulas in FOL.

- A default d is: $\frac{A(X):B(X)}{C(X)}$
- $A(X), B(X), C(X)$ are well-formed formulas
- $X = (x_1, x_2, x_3, \ldots, x_n)$ is a vector of free variables(non-quantified).

Intuitively a default means, **"if $A(X)$ is true, and there is no evidence that $B(X)$ might be false, then $C(X)$ can be true"**.

Example (classical logic):

"All birds fly", $\forall X, \mathit{bird}(X) \rightarrow \mathit{fly}(X)$ (chickens, penguins, kiwis. . . !?)

Default Logic [Reiter]

Definition

A default theory is a pair $\Delta = (D, W)$, where D is a set of defaults and W is a set of formulas in FOL.

- A default d is: $\frac{A(X):B(X)}{C(X)}$
- $A(X), B(X), C(X)$ are well-formed formulas
- $X = (x_1, x_2, x_3, \ldots, x_n)$ is a vector of free variables(non-quantified).

Intuitively a default means, **"if $A(X)$ is true, and there is no evidence that $B(X)$ might be false, then $C(X)$ can be true"**.

Example (classical logic):

"All birds fly", $\forall X, bird(X) \rightarrow fly(X)$ (chickens, penguins, kiwis. . . !?)

$$\forall X, bird(X) \wedge \neg chicken(X) \vee \neg penguin(X) \vee \cdots \rightarrow fly(X)$$

Default Logic [Reiter]

Definition

A default theory is a pair $\Delta = (D, W)$, where D is a set of defaults and W is a set of formulas in FOL.

- A default d is: $\frac{A(X):B(X)}{C(X)}$
- $A(X), B(X), C(X)$ are well-formed formulas
- $X = (x_1, x_2, x_3, \ldots, x_n)$ is a vector of free variables(non-quantified).

Intuitively a default means, **"if $A(X)$ is true, and there is no evidence that $B(X)$ might be false, then $C(X)$ can be true"**.

Example (classical logic):

"All birds fly", $\forall X, bird(X) \rightarrow fly(X)$ (chickens, penguins, kiwis. . . !?)

$$\forall X, bird(X) \wedge \neg chicken(X) \vee \neg penguin(X) \vee \cdots \rightarrow fly(X)$$

Example (Default logic):

"Normally, the birds fly", $D = \frac{bird(X):fly(X)}{fly(X)}$

$$W = \{bird(tweety), penguin(tweety) \rightarrow bird(X), penguin(X) \rightarrow \neg fly(X))\}$$

Default Logic [Reiter]

Definition

E^{Δ} is an extension of Δ iff:

- $E_0 = W$ and
- for $i > 0$, $E_{i+1} = Th(E_i) \cup \{C(X) \mid \frac{A(X):B(X)}{C(X)} \in D, A(X) \in E_i \wedge \neg B(X) \notin E^{\Delta}\}$
- $E^{\Delta} = \bigcup_{i=0}^{\infty} E_i$

Property

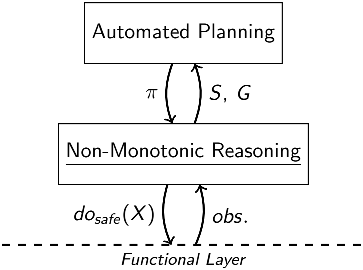
If every default of D is normal: $\frac{A(X):C(X)}{C(X)}$

$$\neg B \notin E^{\Delta} \text{ is replaced by } \neg C \notin E_i$$

there is always one extension and help to perform greedy algorithm

Overview

Contribution



Automated Plan.: FF algorithm

Non-Monotonic Reasoning model

$\Delta = (D, W)$ where:

$D = D_{est} \cup D_{safety} \cup D_{goal}$

$W = W_{obs} \cup W_{est} \cup W_{safety} \cup D_{goal}$

Functional layer: representation of the robot capabilities and features (*skill model) [LesireDG20]*

Notation

observations: obs.; **State and Goal:** S, G; **Sequences of actions:** π; **safe actions applied:** $do_{safe}(X)$

Guidelines of the proposed model

Observations:

From *skill model* (resources states, skill execution statuses)

$W_{obs} = \{at(home) \wedge \neg usbl\_captured \wedge gps\_captured \wedge on\_surface \wedge \cdots\}$

Estimation theory:

Extended knowledge via NMR (non-obvious information)

- $\Delta_{est} = (D_{est}, W_{est})$
  - $d_{loc} = \frac{\top : localized}{localized}$ (localized by default)
  - $\varphi_{loc} = \neg usbl\_was\_captured \wedge \neg gps\_was\_captured \rightarrow \neg localized$
  - $\varphi_{loc'} = low\_loc\_precision \rightarrow \neg localized$

$$d_{loc} \in D_{est}$$

$$\{\varphi_{loc}, \varphi_{loc'}\} \in W_{est}$$

Guidelines of the proposed model

Safety theory:

Emergency rules (**critical failure:** temperature increase, water leakage. . .)

- $\Delta_{safety} = (D_{safety}, W_{safety})$
  - $\frac{A(X):do_{safe}(a)}{do_{safe}(a)}$

$$d_{safe} = \frac{safety\_sensor\_failure(X) : do_{safe}(shut\_down())}{do_{safe}(shut\_down())}$$

$$d_{safe'} = \frac{next\_action(a) : do_{safe}(a)}{do_{safe}(a)}$$

  - $A(X) \rightarrow do_{safe}(a)$ or $A(X) \rightarrow \neg do_{safe}(a)$

$$\varphi_{safe} = low\_energy \rightarrow do_{safe}(shut\_down())$$

$$\varphi_{safe'} = \neg localized \wedge next\_action(transect(X, Y)) \rightarrow \neg do_{safe}(transect(X, Y))$$

$$\{d_{safe}, d_{safe'}\} \in D_{safety}$$

$$\{\varphi_{safe}, \varphi_{safe'}\} \in W_{safety}$$

Guidelines of the proposed model

Goal theory:

Deriving the current mission objective

- $\Delta_{goal} = (D_{goal}, W_{goal})$
  - $\frac{A(X):goal(Y)}{goal(Y)}$ or $\frac{A(X):\neg goal(Y)}{\neg goal(Y)}$

$$d_{goal} = \frac{\top : goal(transect\_done(p_A, p_B))}{goal(transect\_done(p_A, p_B))}$$

$$d_{goal'} = \frac{\neg enough\_energy(transect(p_A, p_B)) : \neg goal(transect\_done(p_A, p_B))}{\neg goal(transect\_done(p_A, p_B))}$$

  - $A(X) \rightarrow goal(Y)$ or $A(X) \rightarrow \neg goal(Y)$

$$\varphi_{goal} = \neg localized \rightarrow \neg goal(transect\_done(X, Y))$$

$$\varphi_{goal'} = close\_seabed \rightarrow goal(on\_surface)$$

$$\{d_{goal}, d_{goal'}\} \in D_{goal}$$

$$\{\varphi_{goal}, \varphi_{goal'}\} \in W_{goal}$$

Guidelines of the proposed model

Programming syntax used

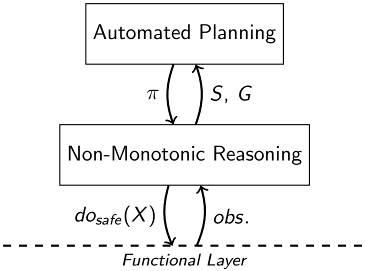
```
%------------------ Language
% on_surface                 : remi is on surface (T) or in water (F)
% low_loc_precision          : localization precision is too low
% low_loc_precision_transect: localization precision is too low for transect
% timeout_task(X)            : X an action; action X has just timed out
%                             (then aborted by the functional layer)
% low_energy                 : energy under a safety threshold
% enough_energy(X)           : X an action; energy is sufficient to perform X
% collision                  : collision (T / F)
% detected(X)                : specie X has been detected
% robot_task(X)              : X an action; robot is performing action X
% next_task(X)               : X an action; the planner has returned X as next action
% transect_done(X, Y)        : X, Y start/end locations
% at(X)                      : X robot location
% timeout_mission            : timeout mission (T / F)

%------------------ ACTIONS
% go_boat
% go_surface
% init_usbl
% transect(X, Y)
% shut_down
% diagnose_motors
```

Guidelines of the proposed model

Programming syntax used

```
%%%%%%%%%%%%%%%%%%% (W_safety)
cl('depth_problem',dur,[sensor_problem(depth)], do_safe(shut_down), 100).
cl('controllability',dur,[-control_actuators], do_safe(shut_down), 100). %(safe4)
cl('SW_mods_active',dur,[-sw_mods], do_safe(shut_down), 100). %(safe5)
cl('problem_motors',dur,[-all_motors_ok], do_safe(diagnose_motors), 100).
cl('collision',dur,[collision], do_safe(shut_down), 100). %(safe8)
cl('low_energy_total',dur,[low_energy], do_safe(shut_down), 100).
cl('not_loc_do_other',dur,[-localized, next_task(transect(_,_))],do_safe(shut_down),100).

%%%%%%%%%%%%%%%%%%% (D_safety)
cl('safety_sensor_problem', def, [safety_sensor_problem(_)], do_safe(shut_down), 100).
cl('do_action', def, [next_task(X)], do_safe(X), 100).

%%%%%%%%%%%%%%%%%%% (W_est)
cl('not_geoloc', dur, [-usbl_was_captured, -gps_was_captured], -localized, 0).
cl('precision_min',dur,[low_loc_precision], -localized, 0).

%%%%%%%%%%%%%%%%%%% (D_est)
cl('loc_by_def', def, [], localized, 0).

%%%%%%%%%%%%%%%%%%% (D_goal)
cl('sensor_problem', def, [sensor_problem(_)], goal(on_surface), 45).
cl('video_problem', def, [sensor_problem(video)], -goal(on_surface), 55).
```

Non-monotonic Reasoning Process

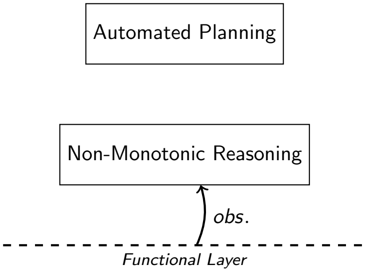


Algorithm NMR process implemented

Require: $D, W \notin \emptyset$
Ensure: $mission\_done \leftarrow \bot$, $W_{obs} \leftarrow \emptyset$

repeat
    $W_{obs} \leftarrow$ obs
    $\Delta = (D, W)$
    $E^{\Delta} = \{S, G, do_{safe}(a)\}$
    if $\exists \{do_{safe}(a)\} \subset E^{\Delta}$ **then**
        `Apply` $do_{safe}(a)$
    else if $\exists \{S, G\} \subset E^{\Delta}$ **then**
        $\pi \leftarrow$ `AutomatedPlan(S,G)`
        if $plan(\pi) = \top$ **then**
            `break`
        end if
        $\Delta = (D, W) \cup \pi$
        $E^{\Delta}_{\pi} = \{S, G, do_{safe}(a)\}$
        if $\exists \{do_{safe}(a)\} \subset E^{\Delta}_{\pi}$ **then**
            `Apply` $do_{safe}(a)$
        else
            $mission\_done \leftarrow \top$
        end if
    end if
until $\neg mission\_done$

Non-monotonic Reasoning Process

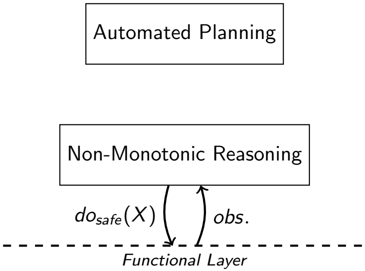


Algorithm NMR process implemented

Require: $D, W \notin \emptyset$
Ensure: $mission\_done \leftarrow \bot$, $W_{obs} \leftarrow \emptyset$

repeat
    $W_{obs} \leftarrow$ obs
    $\Delta = (D, W)$
    $E^{\Delta} = \{S, G, do_{safe}(a)\}$
    if $\exists \{do_{safe}(a)\} \subset E^{\Delta}$ **then**
        `Apply` $do_{safe}(a)$
    else if $\exists \{S, G\} \subset E^{\Delta}$ **then**
        $\pi \leftarrow$ `AutomatedPlan(S,G)`
        if $plan(\pi) = \top$ **then**
            `break`
        end if
        $\Delta = (D, W) \cup \pi$
        $E^{\Delta}_{\pi} = \{S, G, do_{safe}(a)\}$
        if $\exists \{do_{safe}(a)\} \subset E^{\Delta}_{\pi}$ **then**
            `Apply` $do_{safe}(a)$
        else
            $mission\_done \leftarrow \top$
        end if
    end if
until $\neg mission\_done$

Non-monotonic Reasoning Process

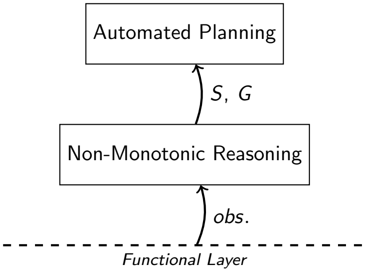


Algorithm NMR process implemented

Require: $D, W \notin \emptyset$
Ensure: $mission\_done \leftarrow \bot$, $W_{obs} \leftarrow \emptyset$

repeat
    $W_{obs} \leftarrow$ obs
    $\Delta = (D, W)$
    $E^{\Delta} = \{S, G, do_{safe}(a)\}$
    if $\exists \{do_{safe}(a)\} \subset E^{\Delta}$ **then**
        `Apply` $do_{safe}(a)$
    else if $\exists \{S, G\} \subset E^{\Delta}$ **then**
        $\pi \leftarrow$ `AutomatedPlan(S,G)`
        if $plan(\pi) = \top$ **then**
            `break`
        end if
        $\Delta = (D, W) \cup \pi$
        $E^{\Delta}_{\pi} = \{S, G, do_{safe}(a)\}$
        if $\exists \{do_{safe}(a)\} \subset E^{\Delta}_{\pi}$ **then**
            `Apply` $do_{safe}(a)$
        else
            $mission\_done \leftarrow \top$
        end if
    end if
until $\neg mission\_done$

Non-monotonic Reasoning Process

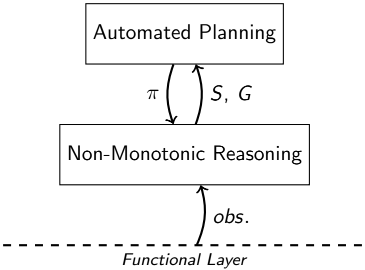


Algorithm NMR process implemented

Require: $D, W \notin \emptyset$
Ensure: $mission\_done \leftarrow \bot$, $W_{obs} \leftarrow \emptyset$

repeat
    $W_{obs} \leftarrow$ obs
    $\Delta = (D, W)$
    $E^{\Delta} = \{S, G, do_{safe}(a)\}$
    if $\exists\ \{do_{safe}(a)\} \subset E^{\Delta}$ **then**
        `Apply` $do_{safe}(a)$
    else if $\exists\ \{S, G\} \subset E^{\Delta}$ **then**
        $\pi \leftarrow$ `AutomatedPlan(S,G)`
        if $plan(\pi) = \top$ **then**
            `break`
        end if
        $\Delta = (D, W) \cup \pi$
        $E_{\pi}^{\Delta} = \{S, G, do_{safe}(a)\}$
        if $\exists\ \{do_{safe}(a)\} \subset E_{\pi}^{\Delta}$ **then**
            `Apply` $do_{safe}(a)$
        else
            $mission\_done \leftarrow \top$
        end if
    end if
until $\neg mission\_done$

Non-monotonic Reasoning Process

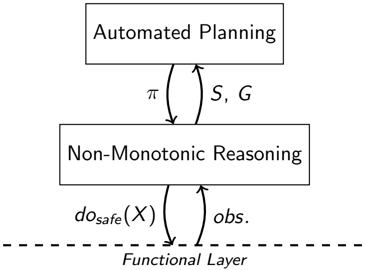


Algorithm NMR process implemented

Require: $D, W \notin \emptyset$
Ensure: $mission\_done \leftarrow \bot$, $W_{obs} \leftarrow \emptyset$

repeat
    $W_{obs} \leftarrow$ obs
    $\Delta = (D, W)$
    $E^{\Delta} = \{S, G, do_{safe}(a)\}$
    if $\exists \{do_{safe}(a)\} \subset E^{\Delta}$ **then**
        `Apply` $do_{safe}(a)$
    else if $\exists \{S, G\} \subset E^{\Delta}$ **then**
        $\pi \leftarrow$ `AutomatedPlan(S,G)`
        if $plan(\pi) = \top$ **then**
            `break`
        end if
        $\Delta = (D, W) \cup \pi$
        $E^{\Delta}_{\pi} = \{S, G, do_{safe}(a)\}$
        if $\exists \{do_{safe}(a)\} \subset E^{\Delta}_{\pi}$ **then**
            `Apply` $do_{safe}(a)$
        else
            $mission\_done \leftarrow \top$
        end if
    end if
until $\neg mission\_done$

Non-monotonic Reasoning Process



Algorithm NMR process implemented

Require: $D, W \notin \emptyset$
Ensure: $mission\_done \leftarrow \bot, W_{obs} \leftarrow \emptyset$

repeat
  $W_{obs} \leftarrow$ obs
  $\Delta = (D, W)$
  $E^{\Delta} = \{S, G, do_{safe}(a)\}$
  if $\exists \{do_{safe}(a)\} \subset E^{\Delta}$ **then**
    `Apply` $do_{safe}(a)$
  else if $\exists \{S, G\} \subset E^{\Delta}$ **then**
    $\pi \leftarrow$ `AutomatedPlan(S,G)`
    if $plan(\pi) = \top$ **then**
      `break`
    end if
    $\Delta = (D, W) \cup \pi$
    $E^{\Delta}_{\pi} = \{S, G, do_{safe}(a)\}$
    if $\exists \{do_{safe}(a)\} \subset E^{\Delta}_{\pi}$ **then**
      `Apply` $do_{safe}(a)$
    else
      $mission\_done \leftarrow \top$
    end if
  end if
until $\neg mission\_done$

Overview

Experimental Procedure

- **Skills interface**[4] (model + manager)
  - operational and descriptive model of the robot
  - 3 data, 9 resources and 10 skills or actions
  - ROS2 middleware (Python/Prolog)
- **Automated Plan.** (FF algorithm)
- **Default Theory** (Prolog)
  - behaviors (goals + safety): modeled with 44 rules (17 normal defaults and 27 exceptions)
  - part of the behaviors came from a risk analysis[5]

Evaluation:

Two simulations were performed, calculation time for a simple as well as a complex problem was evaluated.

[4] Lesire, Charles, David Doose, and Christophe Grand. "Formalization of Robot Skills with Descriptive and Operational Models." IROS, 2020.

[5] Hereau, A., Godary-Dejean, K., Guiochet, J., & Crestani, D. (2021, May). A Fault Tolerant Control Architecture Based on Fault Trees for an Underwater Robot Executing Transect Missions. In International Conference on Robotics and Automation (ICRA 2021).

Simulation and Results

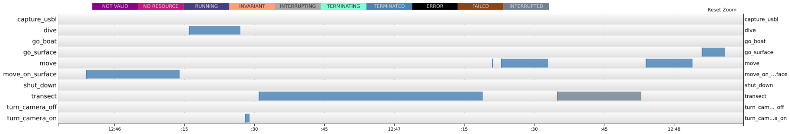


Figure: low localization timeline

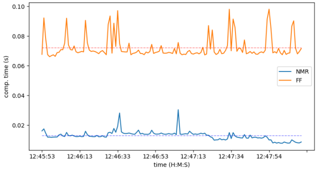


Figure: low localization computation time

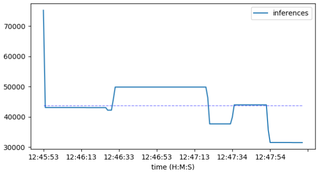


Figure: low localization inferences

Simulation and Results

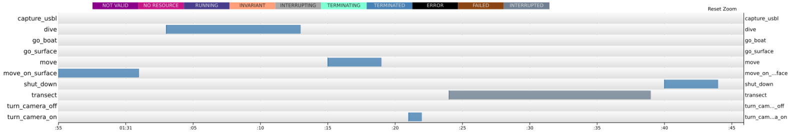


Figure: shutdown timeline

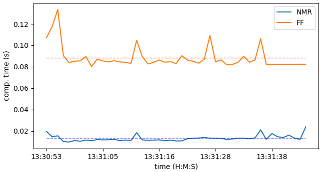


Figure: shutdown computation time

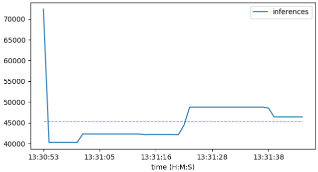


Figure: shutdown inference

Simulation and Results

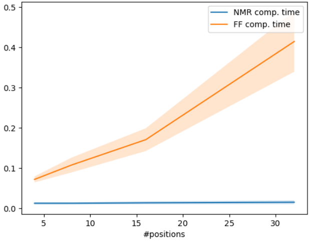


Figure: computation time of 32 positions

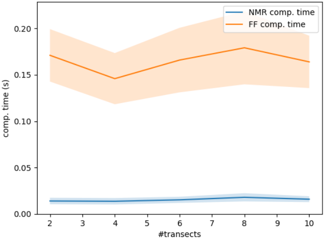


Figure: computation time of goals

Overview

Conclusion

- New decision-making architecture based on a non-monotonic logic:
  - Goal reasoning,
  - Safety rules management
- Default logic is a promising tool for tackling problems that have non-monotonic behavior,
- Model guidelines for use in others applications,
- Practical verification of model complexity (quasi-linear) with 44 rules (17 defaults and 27 exceptions)

Future work:

Multi-agents systems, autonomous agents, other applications. . .

Thank you for your attention,

Questions ?[6]

[6] email: jose.vilchis@ensta-bretagne.fr