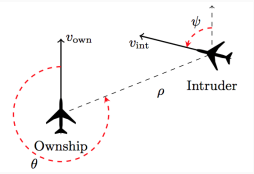
Formuler et prouver des propriétés pour des composants basés IA avec différents prouveurs

Journées GDR GPL LVP-MTV2 2023

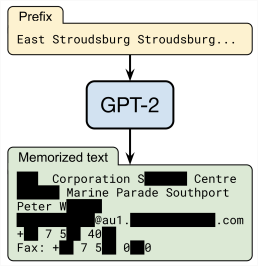
Julien Girard-Satabin (CEA LIST) : julien.girard2@cea.fr
Michele Alberti (CEA LIST) : michele.alberti@cea.fr
François Bobot (CEA LIST) : francois.bobot@cea.fr
Zakaria Chihani (CEA LIST) : zakaria.chihani@cea.fr

Des programmes difficiles à maîtriser...

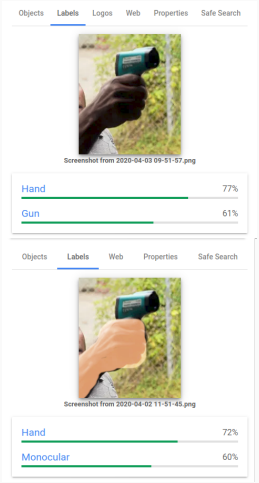
Robustesse locale ?



Propriétés fonctionnelles ?



Respect de la vie privée (en bonus avec les images [Car+23]) ?



Fairness ?

... et beaucoup d'effort pour les analyser

- PyRAT (développé chez nous)
- Marabou [Kat+19]
- Neurify
- ERAN [Sin+19 ; Mül+21]
- $\alpha - \beta$-Crown [Wan+21]
- Nnenum [Bak21]
- NNV (https://github.com/verivital/nnv)
- FaceLattice (https://arxiv.org/abs/2003.01226, https://github.com/verivital/FaceLattice)
- Facet-Vertex incidence (https://github.com/Shaddadi/Facet-Vertex-FFNN)
- Veritex (https://github.com/Shaddadi/veritex)
- Verinet and Venus (https://github.com/vas-group-imperial/VeriNet)

... et beaucoup d'effort pour les analyser

- ReluDiff (https://arxiv.org/abs/2001.03662, https://github.com/pauls658/ReluDiff-ICSE2020-Artifact)
- Peregrinn (https://arxiv.org/abs/2006.10864, https://github.com/rcpsl/PeregriNN)
- Oval (https://github.com/oval-group/oval-bab)
- Libra [Urb+19]
- MIPVerify [TXT19]
- Planet [Ehl17]
- Sherlock [Dut+17]
- ZoPE (https://arxiv.org/abs/2106.05325,https://github.com/sisl/NeuralPriorityOptimizer.jl)
- DNNV [SED21]

(crédit : Bill Wurtz)

Des niches écologique

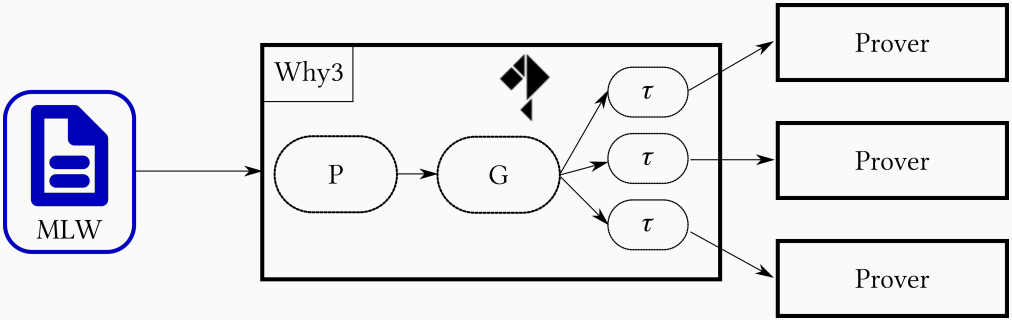
Spécialisations

1. par propriétés : robustesse locale, propriété fonctionnelles, *fairness*;
2. par techniques : Satisfaction Modulo Théorie (SMT), Programmation par Contrainte (CP), interprétation abstraite;

Ne pas se perdre dans la forêt

1. dialogue prouveur-humain difficile : multiples langages et ambiguïtés ;
2. dialogue prouveur-prouveur qui n'existe pas encore ;
3. difficile de savoir quel prouveur utiliser au départ ?
4. modélisation de nouveaux problèmes difficile ;

Why3 à la rescousse

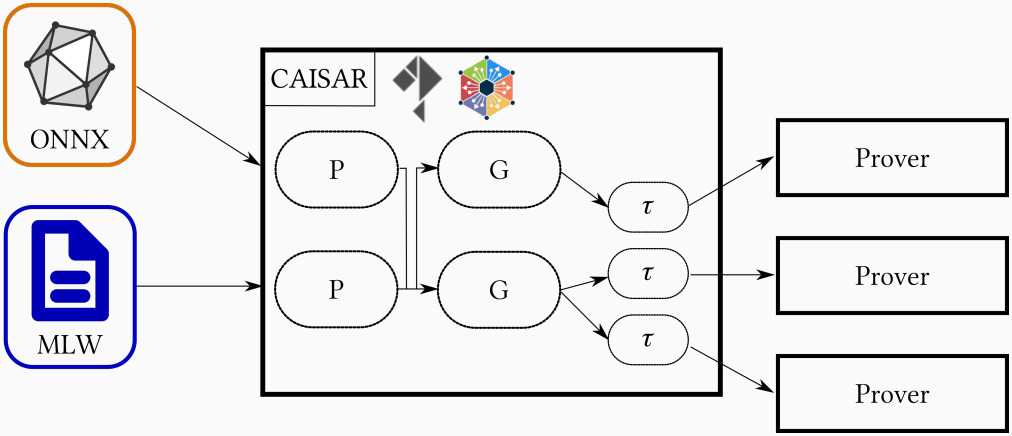


Modélisation unifiée (WhyML), parser (P), générateur d'obligations de preuves (G), transformations (τ)

A software verification platform [FP13]

Website : `https://why3.lri.fr/`

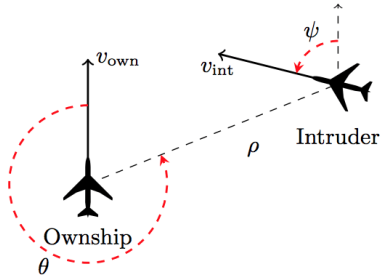
Construire sur du solide



Programme sous différents format (ONNX, NNnet, OVO), gestion de sémantiques de prouveurs différentes

C’est l’heure de la démo :

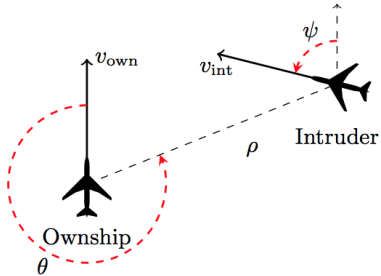
1. décortiquage d’un fichier de spécification ;
2. lancement sur cas d’usage ACAS de plusieurs prouveurs ;
3. examen des spécifications ;



- 5 entrées : ρ, θ, ψ, v_{own}, v_{int}
- 5 sorties : COC, SL, SR, WL, WR

ϕ_1 : « Si l'intrus est directement au dessus et est significativement plus lent que l'appareil, COC sera toujours en dessous d'une certaine valeur »

$\rho \geq 55947,691, v_{own} \geq 1145, v_{int} \leq 60, \text{COC} \leq 1500$



- 5 entrées : ρ, θ, ψ, v_{own}, v_{int}
- 5 sorties : COC, SL, SR, WL, WR

ϕ_3 : « Si l'intrus est directement au dessus et s'approche de l'appareil, COC ne doit pas être le score minimal »

$1500 \leq 1800, -0,06 \leq \theta \leq 0,06, \psi \geq 3,10, v_{own} \geq 980, v_{int} \geq 960$

Spécification

```
theory ACASXU_P3
  predicate valid_input (i: input) = (0.0 :t) .≤ i[distance_to_intruder] .≤ (60760.0 :t)
    ∧ .− pi .≤ i[angle_to_intruder] .≤ pi ∧ .− pi .≤ i[intruder_heading] .≤ pi
    ∧ (100.0 :t) .≤ i[speed] .≤ (1200.0 :t) ∧ (0.0 :t) .≤ i[intruder_speed] .≤ (1200.0 :t)
  [...]
  constant nn_1_1: nn =
    read_neural_network "net.onnx" ONNX
  predicate is_min (o: vector t) (i: int) =
    forall j: int. 0 ≤ j < 5 → i ≠ j → o[i] .≤ o[j]
  goal P3_1_1:
    forall i: vector t.
      has_length i 5 → valid_input i → let is_coc_min = is_min (nn_1_1@@i) 0 in
      ¬ is_coc_min
end
```

Réécritures pour chaque prouveur

```
(**PyRAT, SMTLIB-ish specification **)
;; Goal P3
(assert (<= Y_0 Y_1))
(assert (<= Y_0 Y_2))
(assert (<= Y_0 Y_3))
(assert (<= Y_0 Y_4))

[...]

(**Marabou, prover-specific specification**)
x2 >= 0.49338032358484307238200017309281975030899047851562 5
x3 >= 0.29999999999999998888977697537484345957636833190917968 75
x4 >= 0.29999999999999998888977697537484345957636833190917968 75
+y0 -y1 <= 0
+y0 -y2 <= 0
+y0 -y3 <= 0
+y0 -y4 <= 0
```

Séparation en conjonctions de buts pour les prouveurs qui ne supportent pas la disjonction (oui, ça existe...)

Ce qu'on a ajouté

1. gestion des ensembles d'apprentissage, propriétés non "réellement" universelles ;
2. support de huit prouveurs dédiés : SMT ; interprétation abstraite, test métamorphique ;
3. traducteur du flot de contrôle du programme (réseaux de neurones, SVM, arbres) vers une représentation intermédiaire et du SMTLIB "classique" ;
4. moteur d'interprétation pour faciliter la décharge de preuve aux prouveurs ;

Travaux en cours

1. sémantiques de preuves différentes (tests métamorphiques);
2. granularité dans les tactiques;
3. propagation des résultats de vérification d'un programme vers un autre;
4. explications de décisions;
5. spécifier (et prouver) les données ?

Chantier : extension de l'interpréteur Why3 (*reduction engine*)

Travaux en cours

1. sémantiques de preuves différentes (tests métamorphiques);
2. granularité dans les tactiques;
3. propagation des résultats de vérification d'un programme vers un autre;
4. explications de décisions;
5. spécifier (et prouver) les données ?

Chantier : extension de l'interpréteur Why3 (*reduction engine*)

```
theory MyImportantAIVerif
  use ieee_float.Float64
  use caisar.DatasetClassificationProps

  constant dataset = open_dataset "path/to/dataset"
  constant net = open_program "path/to/onnx"
  constant svm = open_program "path/to/svm"
  constant y = apply net dataset

  goal robustness_svm:
    let eps = (0.5:t) in
    robust svm y eps

  goal functional_prop:
    forall _x in dataset.
      _x[3] .≥ (0.5:t) ∧ _x[2] .≤ (0.0:t) →
      y[3] .≥ 0.5

  (...)

end
```

CAISAR

Site web : `https://caisar-platform.com/`
Logiciel libre (LGPLv2) : `https://git.frama-c.com/pub/caisar`
Rapport technique : `https://hal.science/hal-03687211`
Offres : `https://caisar-platform.com/positions`

Stages, post-docs, CDD à pourvoir ! julien.girard2@cea.fr

La maturation de CAISAR est partiellement financée par les projets PRISSMA et Confiance.ai, du Grand Défi IA de Confiance

Références

[Bak21] Stanley Bak. “Nnenum : Verification of ReLU Neural Networks with Optimized Abstraction Refinement”. In : *NASA Formal Methods*. Sous la dir. d’Aaron Dutle, Mariano M. Moscato, Laura Titolo, César A. Muñoz et Ivan Perez. Lecture Notes in Computer Science. Cham : Springer International Publishing, 2021, p. 19-36. isbn : 978-3-030-76384-8. doi : 10.1007/978-3-030-76384-8_2 (cf. p. 3).

Bibliography ii

[Car+23] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito et Eric Wallace. *Extracting Training Data from Diffusion Models*. 2023. doi : 10.48550/ARXIV.2301.13188. url : https://arxiv.org/abs/2301.13188 (cf. p. 2).

[Dut+17] Souradeep Dutta, Susmit Jha, Sriram Sanakaranarayanan et Ashish Tiwari. "Output Range Analysis for Deep Neural Networks". In : *arXiv:1709.09130 [cs, stat]* (sept. 2017). arXiv : 1709.09130 [cs, stat] (cf. p. 4).

Bibliography iii

[Ehl17] Ruediger Ehlers. “Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks”. In : *arXiv:1705.01320 [cs]* (mai 2017). arXiv : 1705.01320 [cs] (cf. p. 4).

[FP13] Jean-Christophe Filliâtre et Andrei Paskevich. “Why3 - Where Programs Meet Provers”. In : *Programming Languages and Systems*. Sous la dir. de Matthias Felleisen et Philippa Gardner. Lecture Notes in Computer Science. Berlin, Heidelberg : Springer, 2013, p. 125-128. isbn : 978-3-642-37036-6. doi : 10.1007/978-3-642-37036-6_8 (cf. p. 8).

Bibliography iv

[Kat+19] Guy Katz, Derek A. Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljić, David L. Dill, Mykel J. Kochenderfer et Clark Barrett. “The Marabou Framework for Verification and Analysis of Deep Neural Networks”. en. In : *Computer Aided Verification*. Sous la dir. d’Isil Dillig et Serdar Tasiran. T. 11561. Cham : Springer International Publishing, 2019, p. 443-452. isbn : 978-3-030-25539-8 978-3-030-25540-4. (Visité le 18/07/2019) (cf. p. 3).

Bibliography v

[Mül+21] Christoph Müller, François Serre, Gagandeep Singh, Markus Püschel et Martin Vechev. “Scaling Polyhedral Neural Network Verification on GPUs”. In : *Proceedings of Machine Learning and Systems* 3 (2021) (cf. p. 3).

[SED21] David Shriver, Sebastian Elbaum et Matthew B. Dwyer. “DNNV : A Framework for Deep Neural Network Verification”. In : *Computer Aided Verification*. Sous la dir. d’Alexandra Silva et K. Rustan M. Leino. Lecture Notes in Computer Science. Cham : Springer International Publishing, 2021, p. 137-150. isbn : 978-3-030-81685-8. doi : 10.1007/978-3-030-81685-8_6 (cf. p. 4).

Bibliography vi

[Sin+19] Gagandeep Singh, Rupanshu Ganvir, Markus Püschel et Martin Vechev. “Beyond the Single Neuron Convex Barrier for Neural Network Certification”. In : *Advances in Neural Information Processing Systems 32*. Sous la dir. de H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox et R. Garnett. Curran Associates, Inc., 2019, p. 15098-15109. url : http://papers.nips.cc/paper/9646-beyond-the-single-neuron-convex-barrier-for-neural-network-certification.pdf (visité le 27/07/2020) (cf. p. 3).

Bibliography vii

[TXT19] Vincent Tjeng, Kai Xiao et Russ Tedrake. “Evaluating Robustness of Neural Networks with Mixed Integer Programming”. In : International Conference on Learning Representations (ICLR). 2019. url : https://openreview.net/pdf?id=HyGIdiRqtm (visité le 19/06/2019) (cf. p. 4).

[Urb+19] Caterina Urban, Maria Christakis, Valentin Wüstholz et Fuyuan Zhang. “Perfectly Parallel Fairness Certification of Neural Networks”. In : *arXiv:1912.02499 [cs]* (déc. 2019). arXiv : 1912.02499 [cs] (cf. p. 4).

Bibliography viii

[Wan+21] Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh et J. Zico Kolter. *Beta-CROWN : Efficient Bound Propagation with Per-neuron Split Constraints for Complete and Incomplete Neural Network Robustness Verification*. 31 oct. 2021. arXiv : 2103.06624 [cs, stat]. URL : http://arxiv.org/abs/2103.06624 (visité le 04/03/2022) (cf. p. 3).