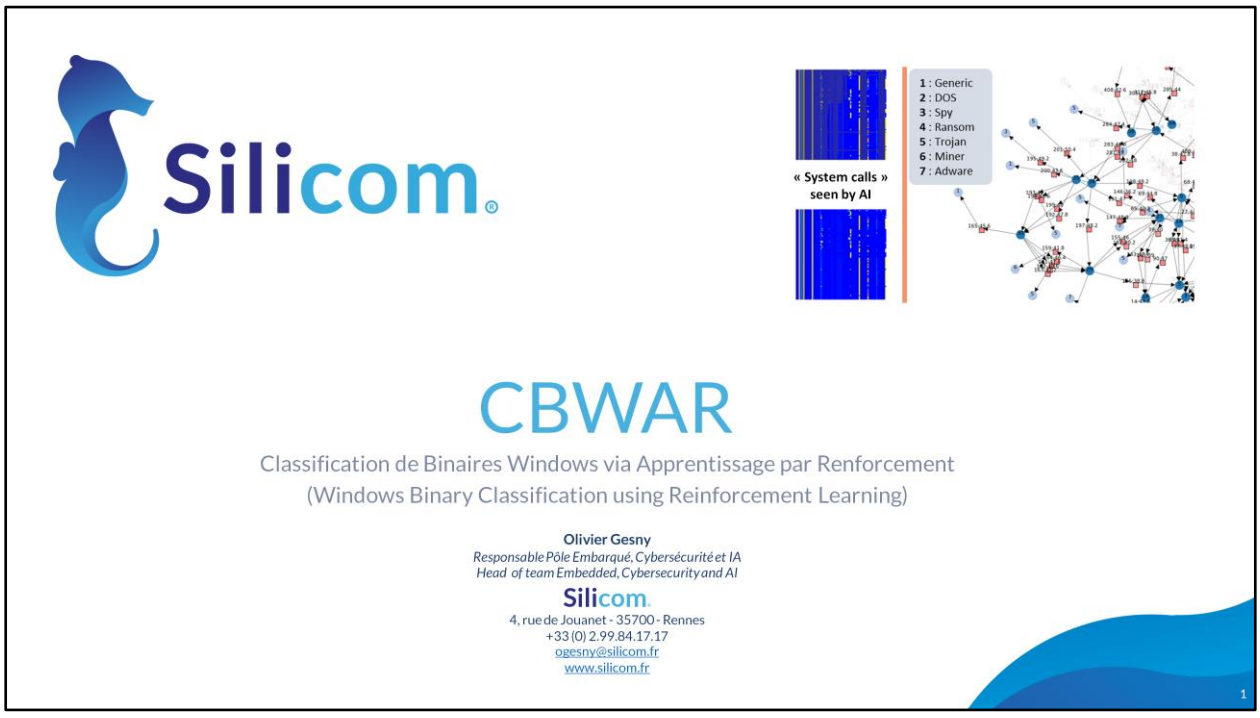
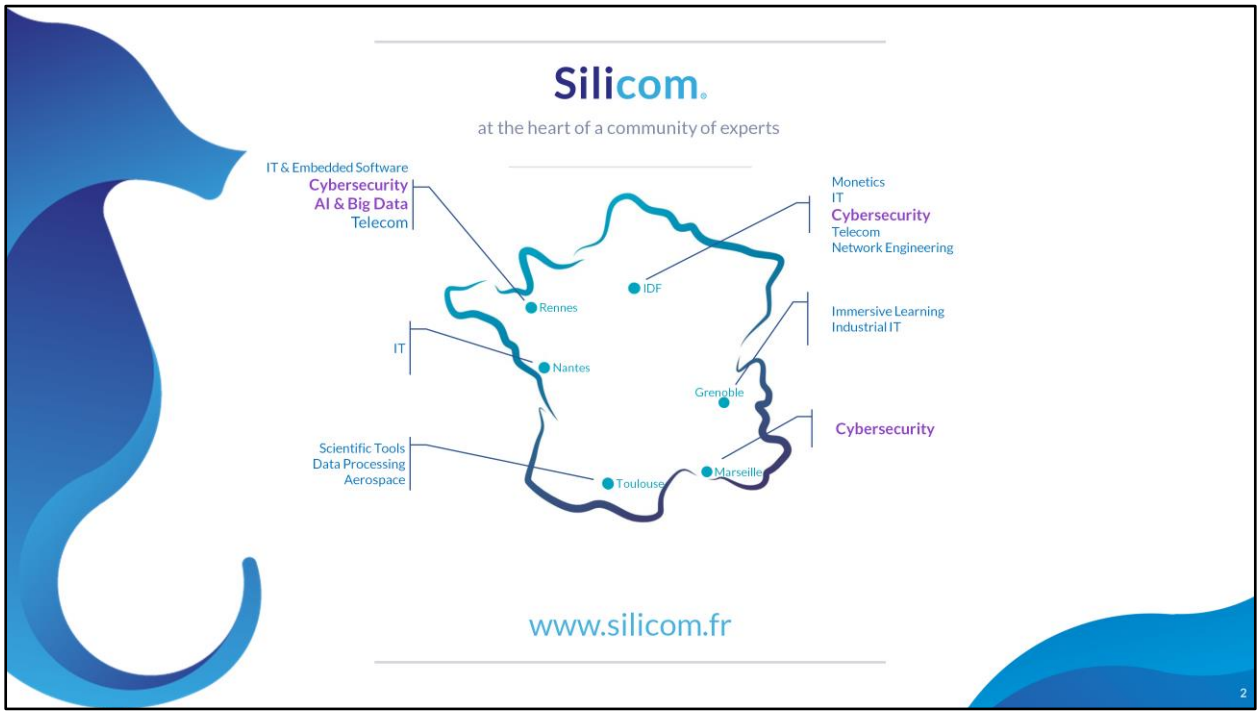


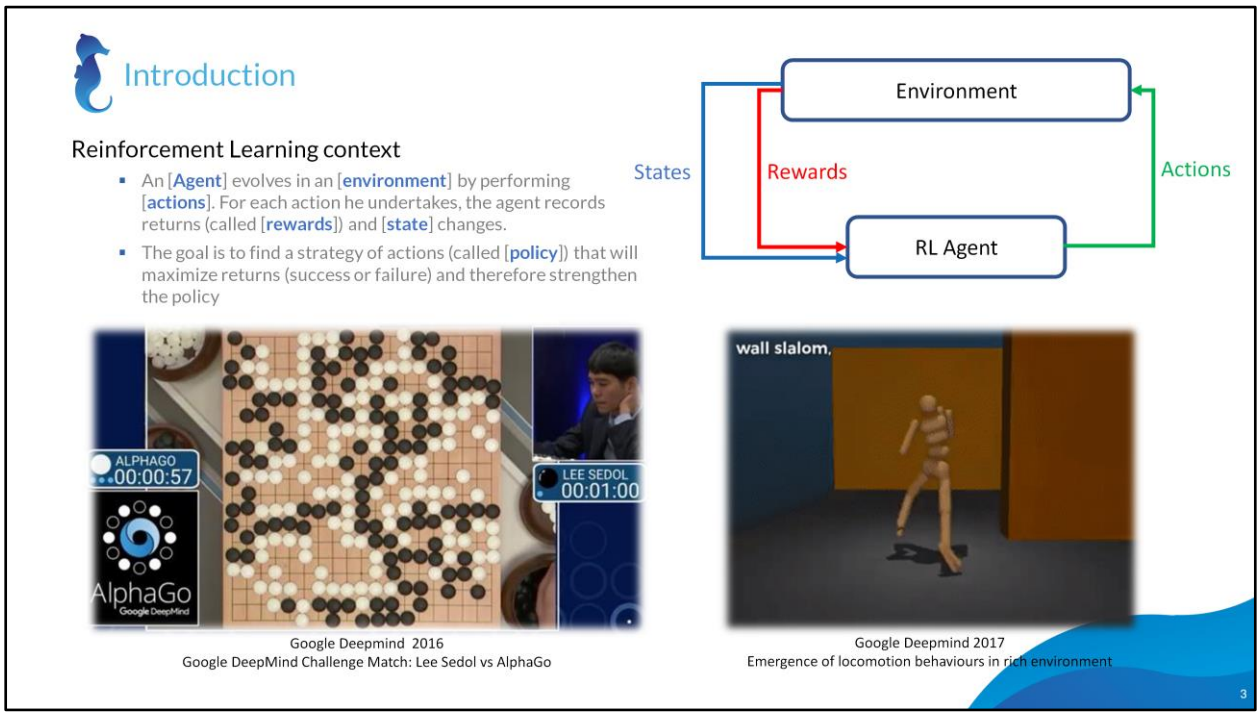
Bonjour à tous.
Je vais vous présenter notre modèle IA qui fait émerger automatiquement des programmes utiles à l'objectif fixé et qui utilise 3 techniques :
- Apprentissage par Renforcement (Reinforcement Learning)
- Programmation génétique (Genetic Programming)
- Et une dose de raisonnement humain (Human Reasonning).

Je vais aussi vous présenter un de ces cas d'application :
- Celui de la classification multiclasse de binaires Windows par analyse de traces d'appels système.

Nous avons initié ce projet avec DGA MI il y a près de 2 ans dans le cadre de la montée en puissance de l'utilisation des techniques IA appliqués à la Cyberdéfense.



Chez Silicom, nous avons plusieurs sites répartis en France dont 3 qui exercent des activités Cyber : en Ile de France, dans le Sud Est (à Toulon essentiellement) et dans l'Ouest (à Rennes) où nous adressons également des problématiques IA et Big Data



Avant d'aborder notre projet, je vais rappeler les principes du Reinforcement Learning :

- Un agent Reinforcement Learning évolue dans un environnement
- Il effectue des actions qui peuvent modifier l'environnement
- Il constate des changements d'état de l'environnement et récupère des récompenses
- L'objectif pour l'agent est de trouver la meilleure stratégie d'actions, celle qui conduit à maximiser les récompenses
- Une stratégie d'action est appelée une politique

Un modèle RL apprend de ses essais, réussites et erreurs pour améliorer la meilleure politique.
Un modèle RL est dans ses fondements un algorithme semi-supervisé dans le sens où il a besoin de données labellisées par l'humain pour apprendre.

Des algorithmes récents basés sur des concepts Reinforcement Learning ont eu des succès retentissants :

- Le joueur virtuel de Go AlphaGo puis AlphaGoZero (de Google Deepmind) qui est capable d'imaginer des stratégies victorieuses jamais envisagées par l'humain et surpasse en quelques heures d'apprentissage les meilleurs joueurs de Go au monde

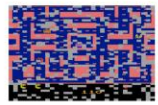
- Et un projet (de Google Deepmind également) qui a bien fait rire la communauté l'an dernier : celui qui a permis de faire émerger une stratégie de locomotion chez un personnage virtuel. Je vous laisse regarder le résultat

Related work

TPG (Tangled Program Graphs)

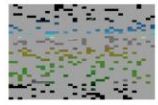
- Best paper at Euro GP 2017
- Best paper at ACM GECCO 2017

Ms. Pac-Man Screen



Ms. Pac-Man AVF

Battle Zone Screen



Battle Zone AVF

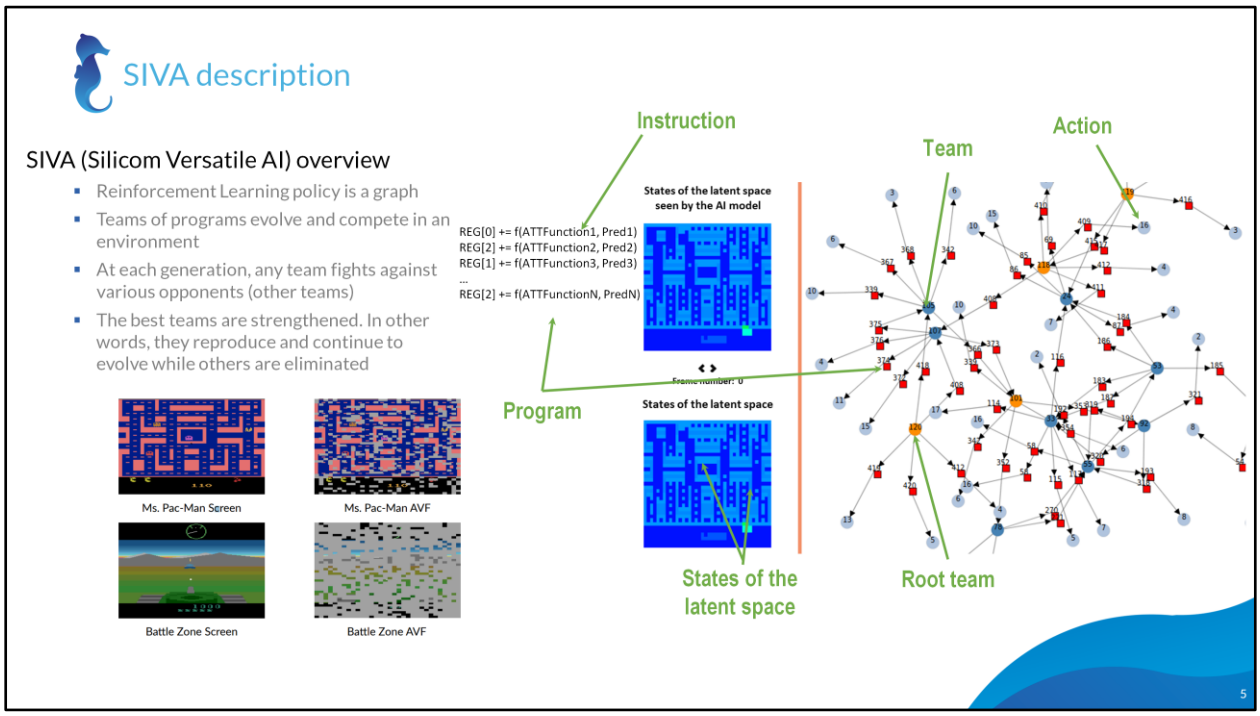
Game	DQN	HNEAT	Hum	TPG	Tms	Ins	%IP
Alien	3069(±1093)	1586	6875	**3382.7**(±1364)	46	455	56
Amidar	**739.5**(±3024)	184.4	1676	398.4(±91)	63	812	69
Asterix	**6012**(±1744)	2340	8503	2400(±505)	42	414	51
Asteroids	1629(±542)	1694	13157	**3050.7**(±947)	13	346	23
BankHeist	429.7(±650)	214	734.4	**1051**(±56)	58	572	65
BattleZone	26300(±7725)	36200	37800	**47233.4**(±11924)	4	123	11
Bowling	42.4(±88)	135.8	154.8	**223.7**(±1)	56	585	57
Centipede	8309(±5237)	25275.2	11963	**34731.7**(±12333)	28	516	39
C.Command	6687(±2916)	3960	9882	**7010**(±2861)	51	280	58
DoubleDunk	-18.1(±2.6)	2	-15.5	2(±0)	4	116	6
Frostbite	328.3(±250.5)	2260	4335	**8144.4**(±1213)	21	382	28
Gravitar	306.7(±223.9)	370	2672	**786.7**(±503)	13	496	36
M'sRevenge	0	0	4367	0(±0)	18	55	28
Ms.Pac-Man	2311(±525)	3408	15693	**5156**(±1089)	111	1036	83
PrivateEye	1788(±5473)	10747.4	69571	**15028.3**(±24)	59	938	60
RiverRaid	**8316**(±1049)	2616	13513	3884.7(±566)	67	660	64
Seaquest	**5286**(±1310)	716	20182	1368(±443)	22	392	37
Venture	380(±238.6)	NA	1188	**576.7**(±192)	3	165	7
WizardOfWor	3393(±2019)	3360	4757	**5196.7**(±2550)	17	247	31
Zaxxon	4977(±1235)	3000	9173	**6233.4**(±1018)	20	424	33

4

Au démarrage du projet, nous avons réalisé un état de l'art des solutions Deep Learning et Reinforcement Learning appliqués à la LID.
Quelques publications intéressantes en sont ressorties mais pas une pléthore non plus (80 publications).
Cependant, un algorithme mêlant Reinforcement Learning et Genetic Programming a particulièrement attiré notre attention (le TPG – Tangled Program Graphs) :
- De par ses propriétés d'émergences élevées exposées dans la publication
- Et parce que l'algorithme dont il s'inspirait (le SBB) avait obtenu de très bons résultats en matière de détection de flux malveillants ou de détection de botnet.

Cet algorithme TPG a été récompensé en 2017 lors de 2 conférences majeures dédiée aux algorithmes évolutionnistes.

Le TPG a la particularité de faire émerger les états qu'il est bon d'observer pour atteindre l'objectif fixé.



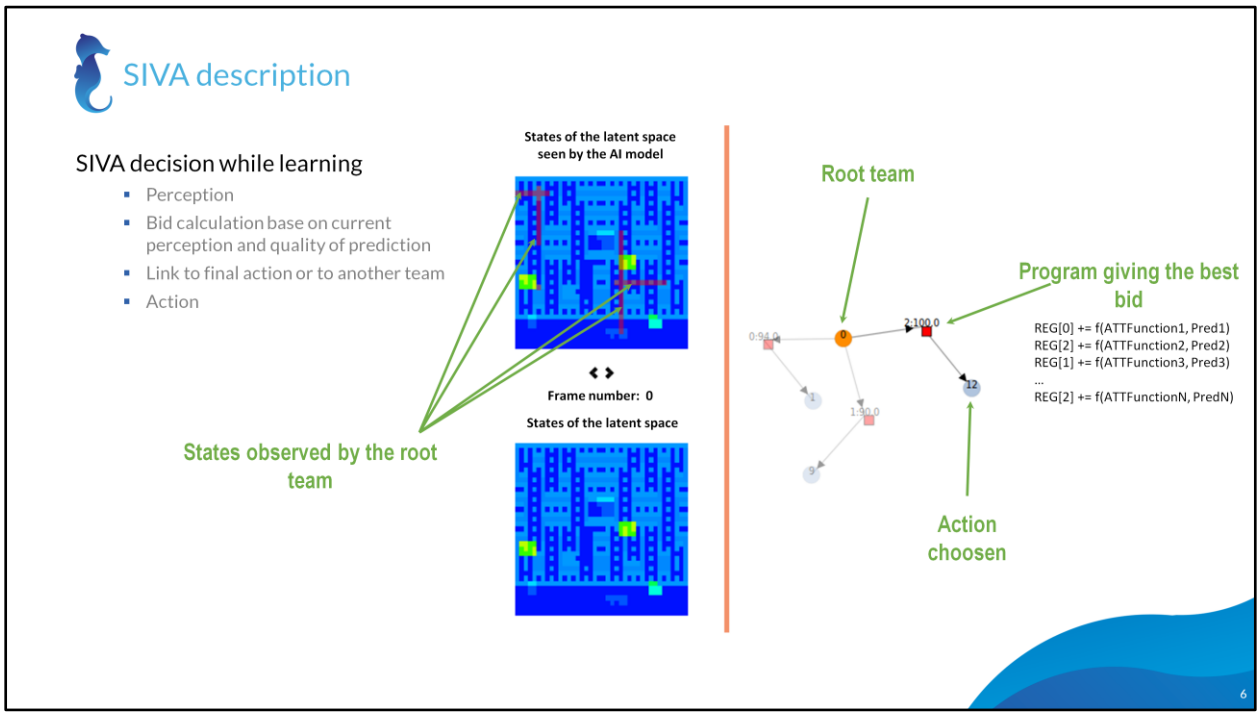
Je vais maintenant vous expliquer les principes de notre algorithme SIVA (qui signifie SIlicom Versatile Artificial Intelligence) et qui utilise des concepts TPG.
La politique est représentée sous la forme d'un graphe.

Ce qu'il faut retenir d'un graphe SIVA :
- Une root team en orange (capitaine) représente le nœud d'entrée d'un arbre de décision
- Une root team est en compétition avec les autres root teams
- Une root team s'appuie sur une multitude d'agents observateurs et décideurs (programmes génétiques) pour prendre une décision d'action finale après chaque analyse de l'environnement
- Un programme est un agent observateur et décideur :
  . Il observe une partie de l'environnement via ses instructions et
  . Il dépose une offre auprès de la team (son chef d'équipe) en s'appuyant sur la qualité de prédiction de ses instructions et sa connaissance acquise ancrée dans les instructions
  . Il pointe vers :
    . soit une action lorsque son offre est finale
    . soit une team standard en bleu foncé (chef d'équipe) pour passer le témoin à d'autres agents qui complèteront l'observation

- Une team standard assure le lien entre les programmes (agents observateurs et décideurs)
- Une instruction d'un programme réalise une seule opération : elle prédit une propriété et calcule cette propriété sur la base :
    . des éléments contenues dans l'environnement courant
    . et d'une fonction d'attention qui se focalise sur le calcul d'une propriété de l'environnement afin de réduire l'espace de décision. Cette fonction d'attention calcule par exemple une propriété de l'environnement telle que un état, un transition d'état temporelle, un nombre [ex : une surface], une distance entre 2 objets, etc
- Une action en bleu clair est par exemple un déplacement unitaire, une décision de classification, l'exécution d'un script, etc

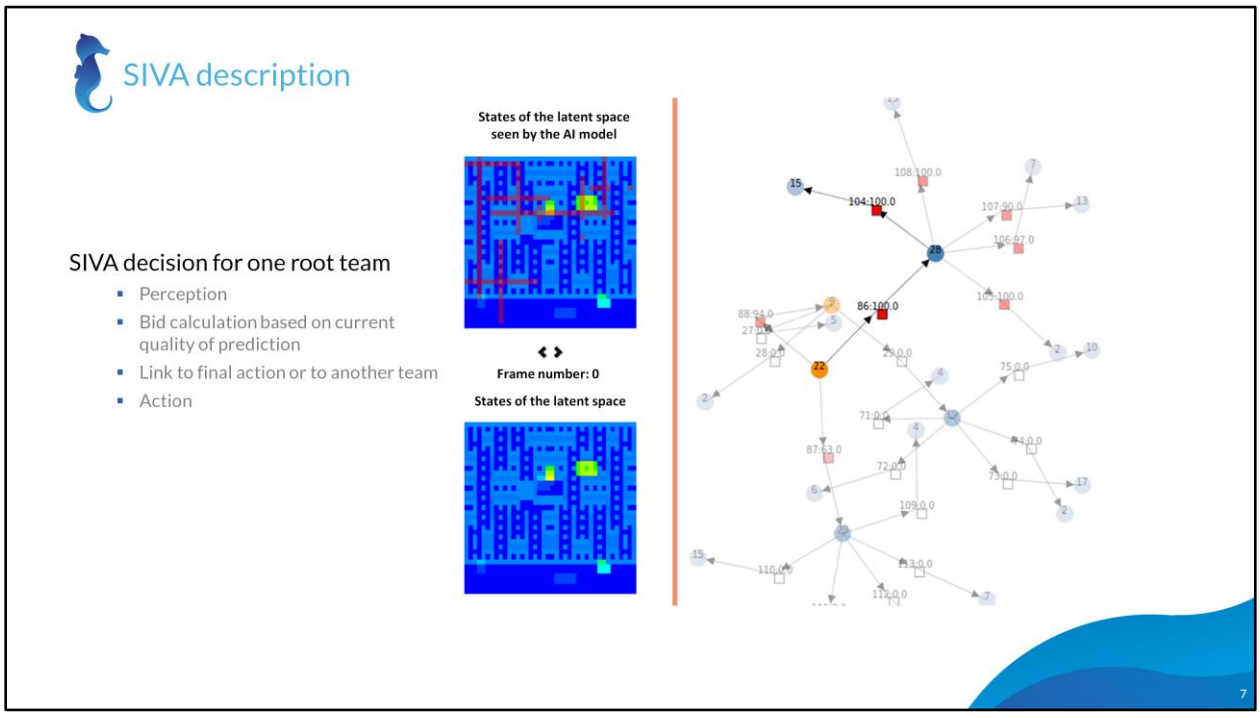
L'idée est de faire coévoluer tous les éléments du graphe pour découvrir progressivement la meilleure politique (meilleure stratégie d'action) pour, par exemple, survivre dans un environnement tout en optimisant ses récompenses, classifier, détecter, optimiser les paramètres d'un algorithme, continuer d'avancer, etc.

La plus value de l'algorithme réside dans :
- Le partage de connaissance via les agents activables par plusieurs root teams
- Sa faculté à ancrer progressivement dans le graphe des corrélations entre observations, actions et objectif à atteindre



Le principe de décision est le suivant :

- La team lance tous les programmes/agents qui exécutent leurs instructions
- Chaque instruction réalise un calcul et le retourne au programme qui calcule la qualité de prédiction de l'instruction
- Sur la base de toutes les qualités de prédiction retournées par les instructions qu'il exécute, le programme retourne son offre à son équipe (team)
- La team choisit le chemin correspondant au programme qui a offert la meilleure offre (comme indiqué sur le schéma où c'est le programme n°2 avec une offre de 100 qui est retenue



La contribution de multiples équipes de programmes rencontrées sur le chemin de l'arbre de décision conduit à l'action finale.

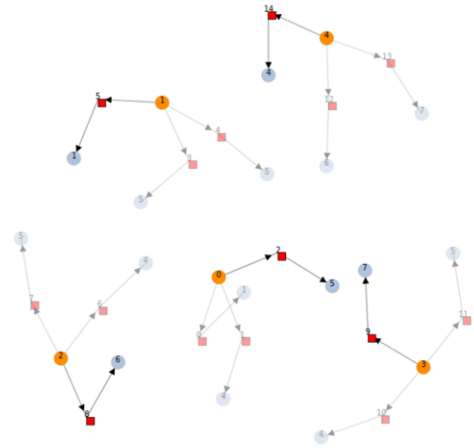
- Chaque équipe (team) suit le chemin du programme qui a rendu la meilleure offre. Une décision est soit une action (ex : classification) soit un passage de témoin à une autre équipe
- Lorsqu'une action est réalisée, une récompense intermédiaire peut être retournée à l'IA. Il peut s'agir d'un écart entre la réalité prédite (par exemple une classification) et la réalité observée

Pour illustrer la prise de décision du TPG/SIVA, voici un petit exemple d'un robot virtuel manipulé par l'IA pour échapper à un autre objet virtuel.
Cette animation montre la dynamique du modèle en termes d'observation et de décision prise.
Avec l'apprentissage, émergent progressivement des instructions qui ont une très bonne qualité de prédiction moyenne et instantanée.

Related work

SIVA evolution while learning

- N root teams in competition
- Each root team executes a batch of challenges of the training dataset
- The worst root teams are eliminated
- The best root team is cloned then mutated



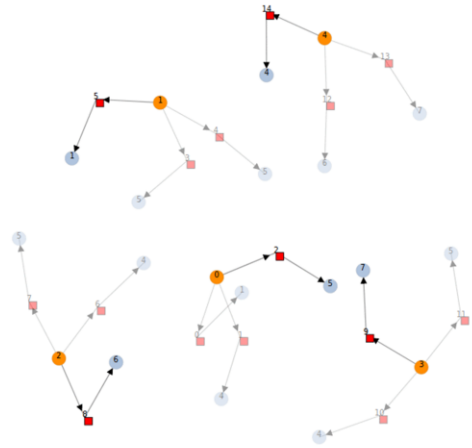
8

Pour la croissance du réseau, le proncipe est qu'à chaque génération :
- N root teams en compétition sur les mêmes challenges d'apprentissage
- Les root teams obtiennent des scores pour chaque challenge.
- Chaque root team est évaluée.
- Les root teams les moins performantes sont supprimées.
- La root team la plus performante est clonée.
- Puis les liens, les programmes et les instructions de l'équipe clonée sont mutés.

Related work

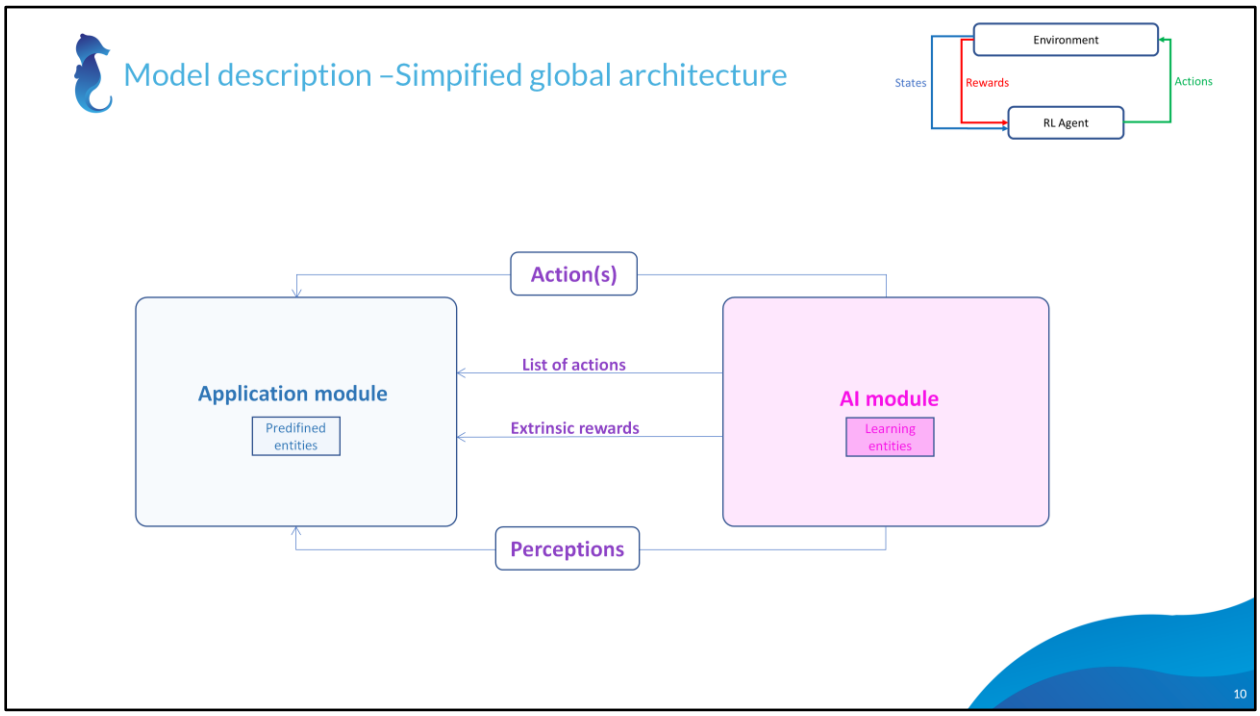
SIVA evolution while learning

- N root teams in competition
- Each root team executes a batch of challenges of the training dataset
- The worst root teams are eliminated
- The best root team is cloned then mutated



9

Le graphe enchevêtré émerge ainsi au fur et à mesure des générations comme on peut le voir sur cette animation montrant l'évolution de notre algorithme SIVA.



Je vais maintenant vous parler plus précisément de notre framework C++.
Dans l'architecture, on distingue 2 parties :
- Le module métier qui gère l'environnement, le système de récompense et les actions décidées par le module IA
- Le module AI (SIVA) basé sur les concepts du TPG

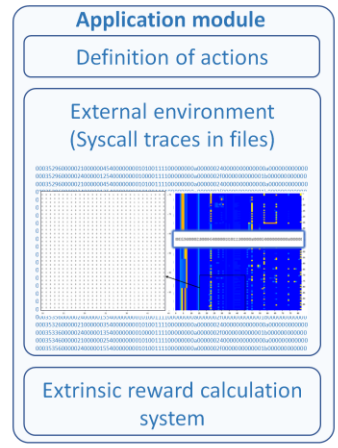
A noter que le module métier est statique alors que les programmes de SIVA évoluent en permanence

Model description – Application module description

Main features

- Latent space building from real environment
- Definition of unitary actions
  - Classification
  - Movement
  - Any other action (script, API call)
- Reward system structuration
  - Application indicators calculation (ex: FPR, TPR)
  - Any positive or negative signal (embodiment)
  - **Score calculation depending on application objective, indicators and other signals**

$$Score = \frac{TP\,.\,PondTP + TN\,.\,PondTN}{FP\,.\,PondFP + FN\,.\,PondFN} \; with \; FP\,.\,PondFP + FN\,.\,PondFN > 0$$



11

Les principales fonctionnalités du module métier sont les suivantes :
- Construction temps réel de l'environnement qui est observé par le module IA (SIVA)
- Définition et exécution des actions choisies par le module IA. Par exemple :
  . des décisions de classification ou d'appartenance à un groupe,
  . des décisions d'équivalence d'identité,
  . des décision de modification de valeur,
  . des mouvements (rotation, déplacement à droite, à gauche, en haut, en bas, avance, recule),
  . des appels de scripts ou API.
- Structuration du système de récompense qui permet de calculer, après chaque action, le score qui dans notre cas l'écart entre la multiclassification rendue par SIVA et la multiclassification réelle.

Le score instantané dépend d'indicateurs métier (par exemple le FPR et le TPR dans notre cas d'application) et traduit l'objectif à atteindre.

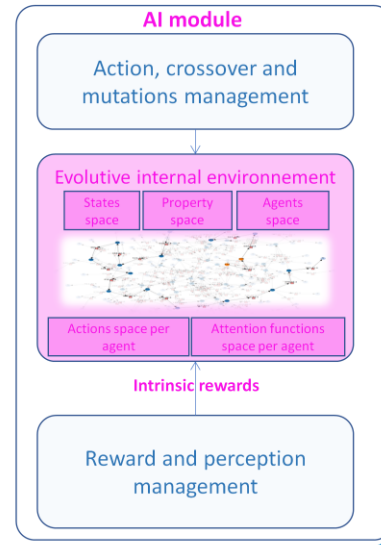
Model description – AI module description

Main features

- Evolutive internal environment
- Instruction and program augmentation regarding TPG
  - Attention functions (distance, states, temporal transitions, number)
  - Instantaneous quality of predictions calculation
  - Live corrections
  - Live mutations
- Anchoring of any nearly confirmed predicted property

Example of innovations

- Predictive agents: perception of an object and comparison with other surrounding objects
- Intrinsic rewards depending on quality of prediction
- Acceptance of predicted property if in the range of acceptable delta to exact property



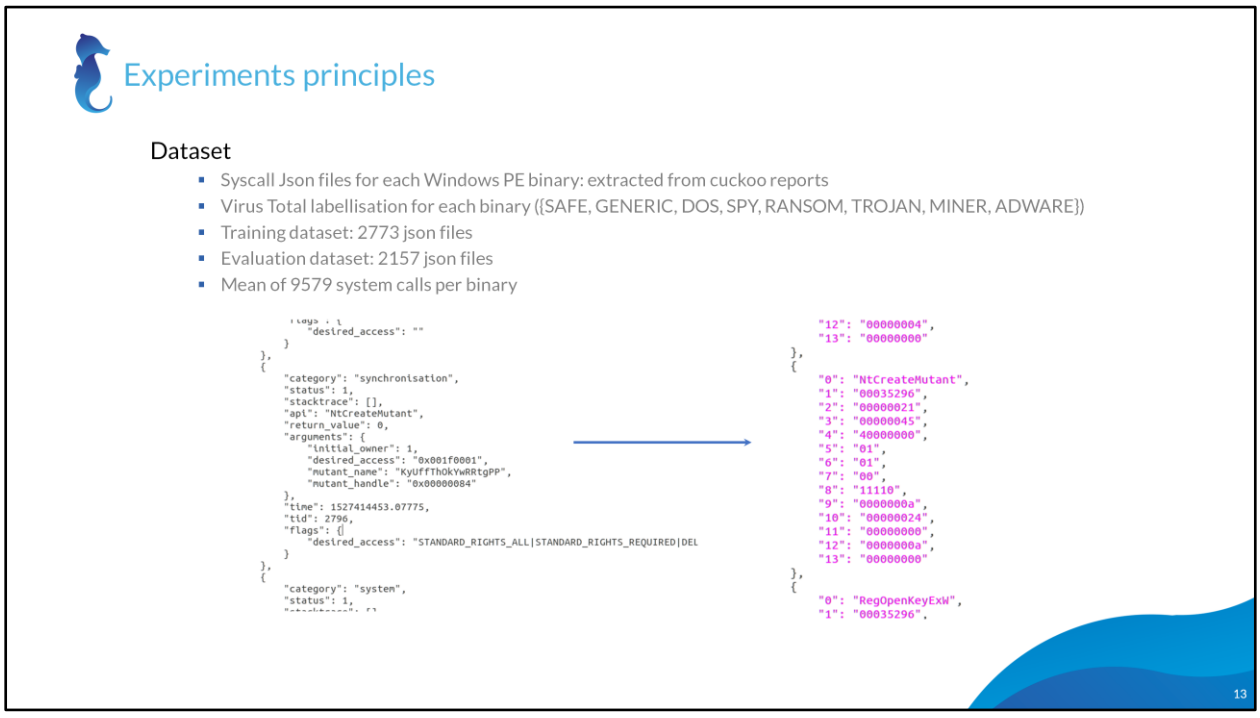
12

Les principales fonctionnalités apportées par SIVA au TPG sont les suivantes :
- Pour les instructions, une dose de raisonnement humain :
. Remplacement d'une opération basique par une fonction d'attention qui prédit une propriété de l'environnement et la calcule sur la base de la dizaine de paramètres définis dans l'instruction
. La fonction d'attention (dose de raisonnement humain) permet de calculer l'état d'un objet, un nombre (par exemple : une surface), sa transition d'état avec le passé, des comparaisons entre 2 objets (états, distances, transition d'état). Ces fonctions d'attention sont hardcodées et on peut en développer facilement de nouvelles
- Pour les programmes/agents :
. La correction de chaque instruction la première fois qu'une propriété est correctement prédite ou presque correctement prédite
. Le programme ancre dans le réseau toute nouvelle prédiction correcte et renforce toute nouvelle prédiction presque correcte
. Le calcul de la qualité de prédiction de chaque instruction qui représente l'écart entre la propriété prédite et la propriété exacte
. Modification de l'offre qui est le reflet de la contribution de la qualité de prédiction de chaque instruction
. L'ajout et la suppression d'instructions après chaque action. Cela permet de faire

évoluer plus rapidement le modèle avec des programmes ayant de très bonnes qualités de prédiction

Toutes ces innovations se traduisent dans les 2 nouveaux espaces puisque ceux-ci émergent au fur et à mesure de l'évolution dans l'environnement :
- Espace des propriétés
- Espace des fonctions d'attention par agent

Constitution d'une instruction :
operation : fonction d'attention
tpgRegister : registre TPG
currentRefAddressInEnvironment : adresse de l'"objet référence" dans l'environnement présent
currentCompAddressInEnvironment : adresse de l'"objet comparé" dans l'environnement présent
elapsedDurationSinceLatestObservation : saut temporel écoulé depuis la dernière observation de l'"objet "
foveaSize : rayon d'observation de la fonction d'attention
currentRefState : état courant à l'adresse *currentRefAddressInEnvironment*
currentCompState : état courant à l'adresse *currentCompAddressInEnvironment*
passedRefState : état passé de l'adresse *currentRefAddressInEnvironment* à *t-elapsedDurationSinceLatestObservation*
passedCompState : état passé de l'adresse *currentCompAddressInEnvironment* à *t-elapsedDurationSinceLatestObservation*
distanceRefComp : distance entre les 2 objets comparés
predictedProperty : propriété prédite
acceptablePercentageOfErrorToExactProperty : pourcentage d'erreur acceptable pour la propriété prédite au regard de la propriété exacte
numberOfAcceptablePredictions : nombre de prédictions acceptables
meanInstructionPredictionQuality : moyenne de la qualité de prédiction de l'instruction. Mise à jour exclusivement lorsque la prédiction est acceptable
numberOfExecutions : nombre d'exécutions de l'instruction



Nous avons entraîné et testé notre modèle dans un but d'identification du type de malveillance de binaires Windows PE 32 bits :

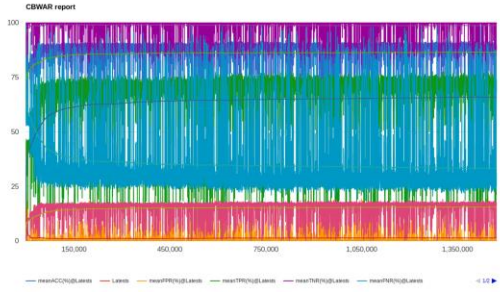
- labellisés par le biais de VirusTotal
- exécutés en environnement sandboxé Cuckoo pour obtenir des rapports contenant des traces d'appels système

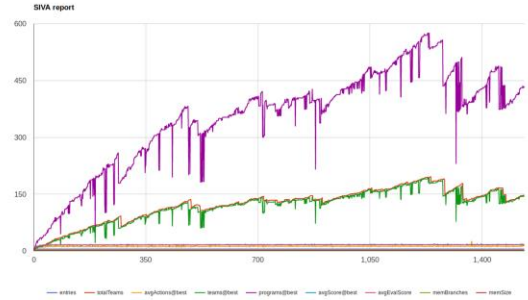
Cela nous a permis de générer 2 dataset de fichier d'appels système JSON (extrait des rapports Cuckoo) :

- un pour l'apprentissage
- un pour l'évaluation

Results

Experiments	Training duration (ms)	FPR (%)	ACC (%)	TPR (%)	TNR (%)	FNR (%)
Acquisition window : 100 Ponderations : - TP : 0.75; FP : 1.0; TN : 0.0; FN : 0.0	113478 (~32h)	0,28	89,16	71,67	99,76	27,12





14

Nous avons paramétré le score dans le module métier de telle sorte de favoriser un objectif de faux positifs (FPR) de 0%.
Après un apprentissage de 32h sur le dataset d'apprentissage, nous obtenons un FPR de 0,28% sur un PC standard comportant 16 Go de RAM et un processeur tournant à 3,4 Ghz
J'ai voulu montrer des courbes d'apprentissage métier pour montrer que le modèle en phase d'apprentissage essaie de nouvelles solutions pour lui permettre de sortir de puits de potentiels (apprentissage par essais, réussites et erreurs).
La courbe d'apprentissage SIVA montre que le nombre d'équipes et de programmes croît puis se stabilise.

Conclusion

Example of added values

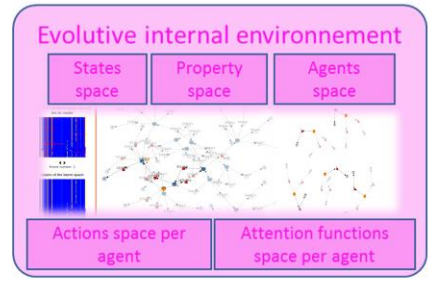
- **Emergence** of a **decision tree**
- Strong **correlation between attention functions, perception of the agents, action and goal to be achieved** thanks to emergence of links between teams, programs and actions
- **Explainability**
- **Discrimination** power
- **Adaptable** to many use cases (times series mining, classification, optimization, etc)

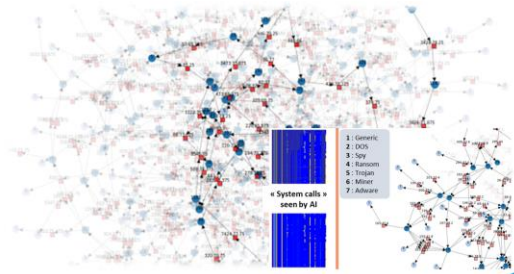
Weakness

- **Mutations** are **still too random** so link between agents emerge not as quickly as it could be
- Not easy to be parallelized
- **Training not** yet as **quick enough** as it could be

Be careful (ethical considerations)

- **Define carefully application objectives considering ethics**
- **Define carefully the unitary actions**: some of them are dangerous if given initially to the learning model => do not give a weapon to a baby





15

Les résultats obtenus sont déjà bons de notre point de vue et le modèle présente quelques plus values notables :

- Feature extraction (Extraction de propriétés) automatique grâce aux fonctions d'attention qui permettent de calculer les distances, états, nombres et transitions caractéristiques des types de malveillance rencontrées => fort pouvoir de segmentation
- De fortes corrélations émergent entre les fonctions d'attention, l'observation des agents, les actions et l'objectif à atteindre
- Explicabilité parce qu'il de suivre le chemin suivi par la root team pour comprendre la décision prise

Quelques faiblesses :

- Les liens d'intérêt n'émergent pas assez rapidement et l'algorithme n'est pas aisé à paralléliser
- Exploiter visuellement les features automatiquement extraites lors de l'émergence et l'évolution des programmes qui sont ancrées dans le graphe

Attention à l'éthique de ce type de modèle car il est aisé de mettre n'importe quelle action entre les mains du modèle qui peut faire des dégâts en phase d'apprentissage du fait des de la logique essais, réussites et erreurs du modèle => Ne mettez pas tout de suite un couteau entre les mains d'un bébé. Nous avons

d’ailleurs été confronté à ce problème puisque nous avons mis des actions de déplacement/clic d’une souris et d’appui touche clavier. Mettre ce type d’action entre les mains de notre IA émergente nous a conduit à éteindre notre PC car celui-ci était devenu incontrôlable.

Life evolution as inspiration for AI improvement

- Organisms behaviours and interactions
- Symbiotic evolution
- Auto organization
- Neurosciences
- Learning cells

https://www.popularmechanics.com/science/a23842/slime-mold-slime-lapse/

16

Nous nous inspirons énormément du vivant pour élaborer nos modèles.
D'ailleurs, ne trouvez vous pas qu'il y a un air de parenté entre SIVA et le blob ? Je vous laisse observer son évolution.

Life evolution as inspiration for AI improvement

- Organisms behaviours and interactions
- Symbiotic evolution
- Auto organization
- Neurosciences
- Learning cells

Curiosity as inspiration for AI improvement

- Oudeyer, Lecun, Bayes, Pearl, Domingo
- Lachaux, Dehaene, Dessalles, Houdé, Sablonnière, Changeux
- Karsenti, Bapteste, Dussutour
- Berthoz, Debru, Jeannerod, Damier
- Ameisen, Klein, Lê Nguyên Hoang

Audrey Dussutour & David Vogel - Pour la Science N°483 - Janvier 2018

Questions?

17

La curiosité dans toutes les disciplines (IA, neurosciences, comportement du vivant, …) est une source d'inspiration pour l'amélioration de nos modèles IA.

Une démo post apprentissage est disponible dans le fichier CBWAR_C&ESAR.mp4.