Version 4.5.0

Date de diffusion : Mars 2019

L'Atelier B 4.5.0 est une version ***Free Edition***, dont l'accès est libre, directement téléchargeable sur le site web Atelier B[1].

Nouvelles fonctionnalités / Caractéristiques

L'Atelier B 4.5.0 a été mis à disposition en Mars 2019.

Cette version corrige, de manière cumulative[2] depuis la version 4.2.1, 146 anomalies et améliore les fonctionnalités suivantes :

- Installation simplifiée sous linux [4.5]
- Génération de PO
  - pour le B événementiel, les obligations de preuve de non-interblocage, non-divergence, faisabilité, exclusivité et couverture sont générées par le nouveau GOP [4.5]
- Preuve automatique
  - outils d'interfaçage « *IAPA* » (sous linux uniquement) vers les prouveurs disponibles avec la plateforme *Why3*[3] [4.5]
- Preuve interactive
  - Introduction des Tâcherons de preuve qui, grâce à l'utilisation de prouveurs SMT, permettent de définir des règles de preuve [4.5]
  - Possibilité d'ajouter ses propres groupes de commande de preuve [4.5]
  - Amélioration du typage des commandes de preuve [4.5]
  - Ajout d'un timeout paramétrable aux commandes de preuve de la famille pp [4.4.2]

[1] https://www.atelierb.eu/telechargement/
[2] Au travers des versions commerciales 4.3.0 et 4.4.2
[3] http://why3.lri.fr/

    - Ajout d'une nouvelle commande de preuve at pour Apply Tactic [4.4.2]
    - Ajout de fonctionnalités dans l'outil de preuve de règles [4.4.2]
    - Intégration du model-checker ProB dans le prouveur interactif [4.3]
- Support des tableaux indexés par des énumérés pour le générateur de code C4B [4.5]
- Affichage des obligations de preuve dans l'éditeur [4.4.2]
- Vérification des règles de codage au sein de l'Atelier B [4.3]

Les fonctionnalités ajoutées aux versions 4.4.2 et 4.3 sont décrites dans leurs notes de version respectives (notes de version 4.4.2, notes de version 4.3).

Installation linux Universal

Il s'agit d'un fichier zip que l'utilisateur peut désarchiver à n'importe quel endroit sur son disque dur, afin de faciliter le processus d'installation.

Une fois désarchivé, ll faut modifier manuellement 3 fichiers : ***AtelierB***, ***startAB*** et ***startBB***.

Ces fichiers contiennent des chemins par défaut, qu'il faut adapter en fonction du répertoire d'installation sur le système cible.

Dans ***AtelierB*** il faut changer les lignes suivantes :

```
ATB*ATB*AtelierB_Directory: /<repertoire_installation>/atelierb-free-4.5.0/
ATB*ATB*Atelier_Database_Directory: /<repertoire_installation>/atelierb-free-4.5.0/press/bdb
ATB*ATB*Print_Command: /<repertoire_installation>/atelierb-free-4.5.0/bbin/bprint
ATB*BART*RefinerFile: /<repertoire_installation>/atelierb-free-4.5.0/press/include/PatchRaffiner.rmf
```

Dans ***startAB*** il faut changer les lignes suivantes:

```
LD_LIBRARY_PATH/<repertoire_installation>/atelierb-free-4.5.0/bbin/linux_x64:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
/<repertoire_installation>/atelierb-free-4.5.0/bbin/linux_x64/AtelierB $*
```

Dans ***startBB*** il faut changer les lignes suivantes:

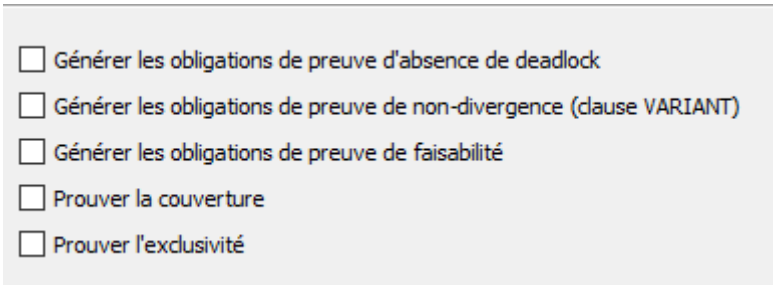
```
LD_LIBRARY_PATH=/<repertoire_installation>/atelierb-free-4.5.0/bbin/linux_x64:$LD_ LIBRARY_PATH
export LD_LIBRARY_PATH
/<repertoire_installation>/atelierb-free-4.5.0/bbin/linux_x64/bbatch -r=/<repertoire_installation>/atelierb-
free-4.5.0/AtelierB $*
```

Génération d'obligation de preuve en B événementiel

Le fichier de paramétrage des PO pour le B événementiel (paramGOPSystem.xsl) a été enrichi des obligations de preuves suivantes :

- non-interblocage (DLF),
- non-divergence (DIV),
- faisabilité (FIS),
- couverture (COV),
- exclusivité (EXC)

Pour les générer, il faut activer les différentes options dans le menu *préférences/modélisation du système* du projet.



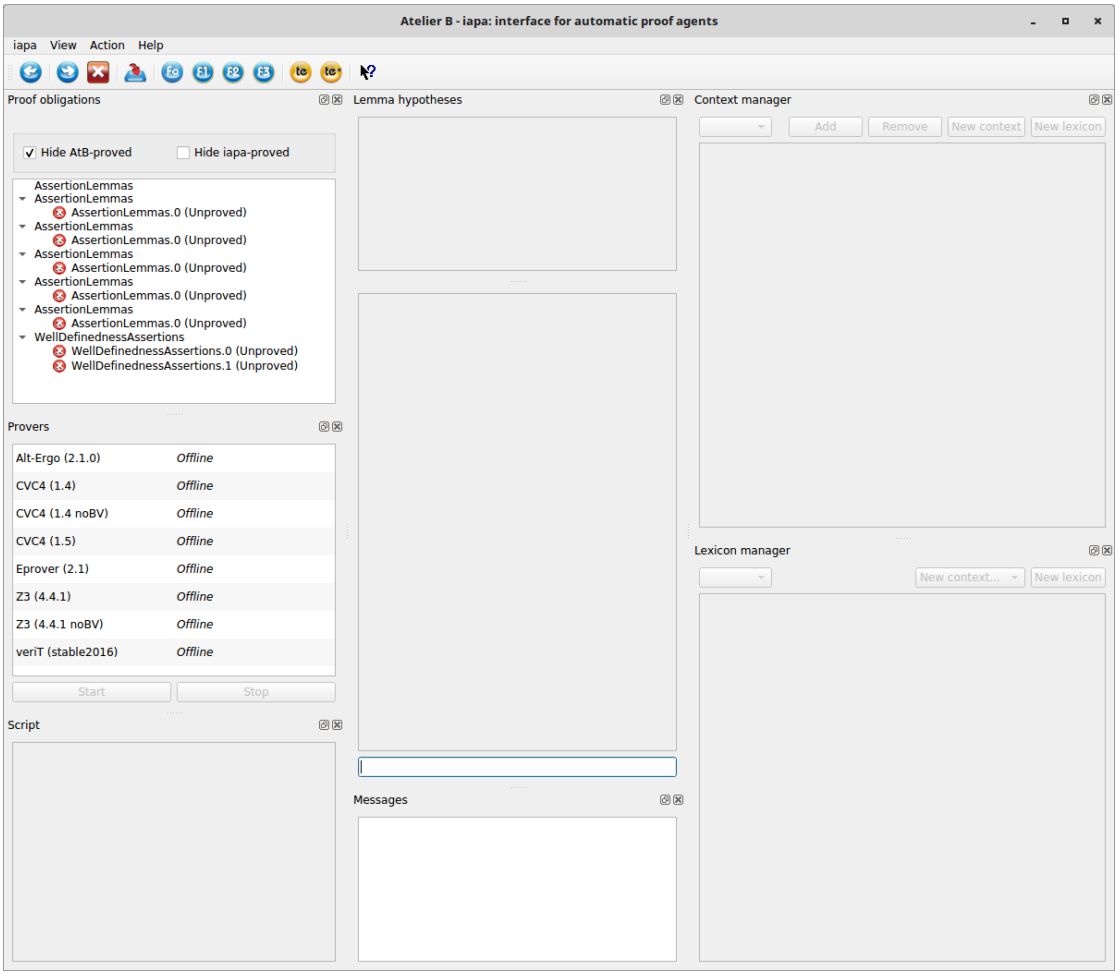
De plus le principe des témoins (WITNESS) a été rajouté au GOP NG. Ces témoins permettent de faciliter la preuve de faisabilité de substitution non-déterministe en fournissant une valeur pour chaque variable quantifiée (voir exemple ci-dessous).

```
SYSTEM W0
VARIABLES
    xx
INVARIANT
    xx : NATURAL
INITIALISATION
    xx := 0
EVENTS
    EE = ANY zz WHERE zz : 0 ..9 THEN xx := zz END
END
```

```
REFINEMENT W0_r
   REFINES W0
VARIABLES
    xx
INVARIANT
    xx : NATURAL
INITIALISATION
    xx := 0
EVENTS
    EE = BEGIN WITNESS zz = 1 THEN xx := 1 END END
END
```

Outil IAPA (Linux)

Le nouveau module *IAPA* (Interface for Automatic Proof Agents) de l'Atelier B est une interface de preuve « presque automatique » mettant à profit les solveurs accessibles au travers de la plateforme Why3 (travaux issus du projet collaboratif BWare).

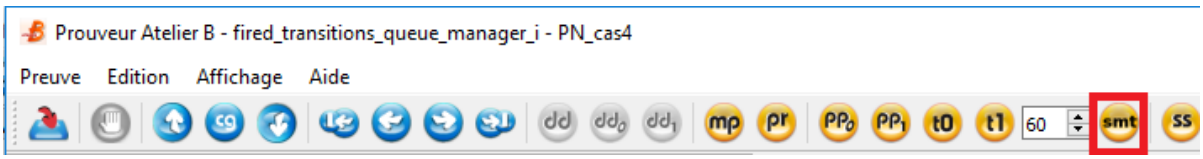


C'est une interface automatique dans le sens où le but des obligations de preuves (OP) est envoyé tel quel aux prouveurs. Il est néanmoins demandé de sélectionner les hypothèses qui y sont adjointes. En effet les obligations de preuve (OP) complètes des projets industriels réels peuvent contenir beaucoup d'hypothèses qui ne sont pas toutes en rapport avec la démonstration d'une OP particulière. Ce grand nombre de formules peut fortement ralentir les prouveurs, voir les empêcher de fonctionner correctement.

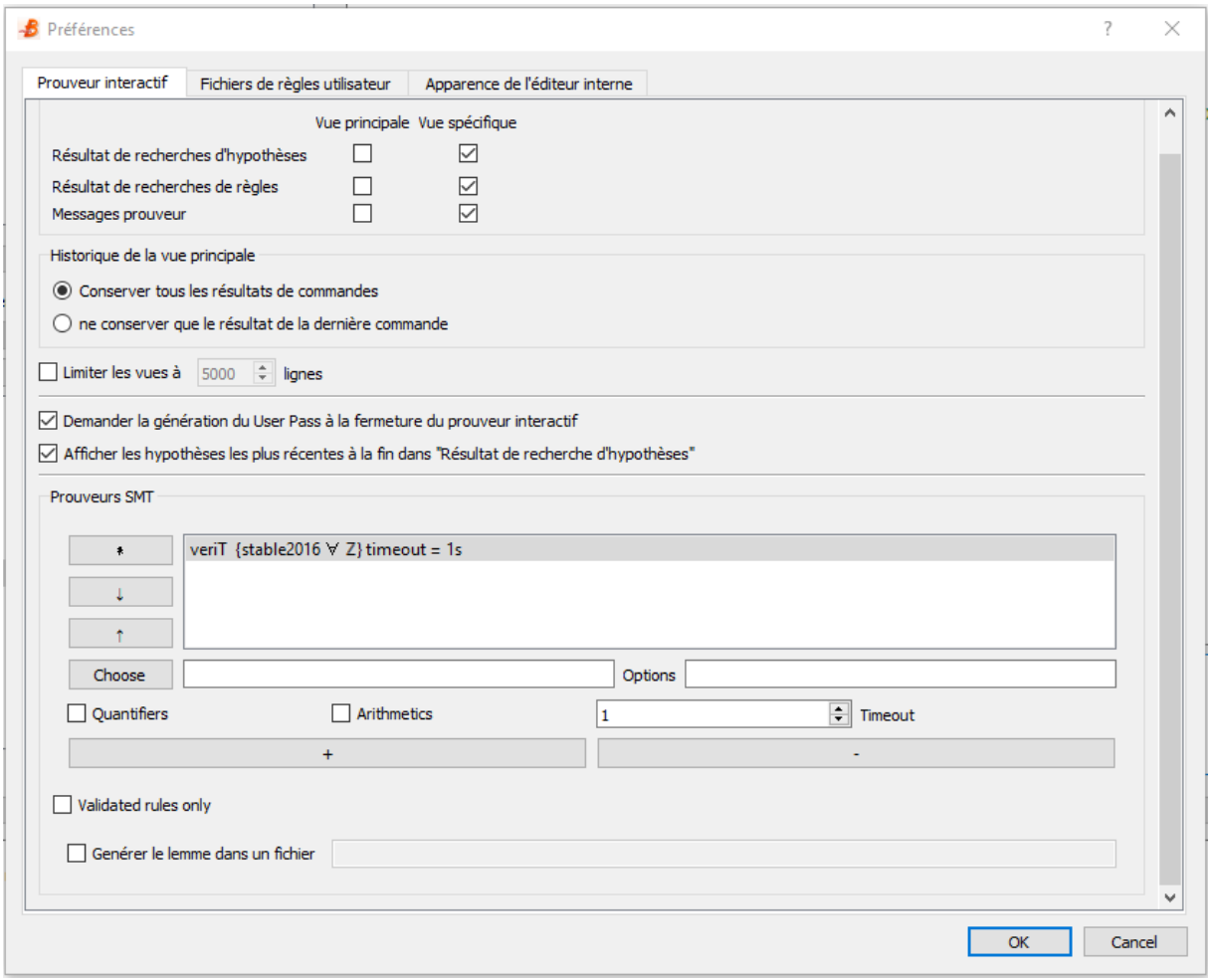
Il s'agit donc d'un nouvel outil interactif (séparé des outils habituels) qui présente le but et les hypothèses d'une OP et qui permet de lancer les différences prouveurs externes sur une sélection d'hypothèses associée à chaque but. Divers mécanismes sont fournis afin de pouvoir réaliser cette sélection de manière efficace et reproductible. On peut considérer *IAPA* comme une approche à mi-chemin entre les outils automatiques (F0, F1, …) de l'Atelier B et celui de preuve interactive.

Outils tâcherons

La fonction « tâcherons » est un nouveau mécanisme qui vient enrichir la preuve interactive existante de l'AtelierB. Elle permet d'automatiser la démonstration d'une étape intermédiaire.



Cette fonction s'utilise par une nouvelle commande de preuve qui tente de décharger le but courant en exécutant des solveurs externes (de type SMT). Si un de ces solveurs réussit, une règle de preuve est alors produite, qui capture la forme du théorème correspondant au but courant. Cette règle est sauvegardée avec les autres règles de preuve et peut être réutilisée à volonté et ceci même sur un poste de travail ne disposant pas des solveurs externes.



L'avantage principal des tâcherons est qu'ils permettent d'exploiter les capacités des solveurs SMT tout en restant compatible avec le processus habituel de preuve de l'AtelierB. Il est même possible de vérifier en sus les règles produites par les tâcherons, avec les outils de preuve de règles de l'Atelier B.

Afin de pouvoir utiliser les prouveurs SMT, il faut dans un premier temps installer un prouveur SMT4,5,6 et configurer les préférences.

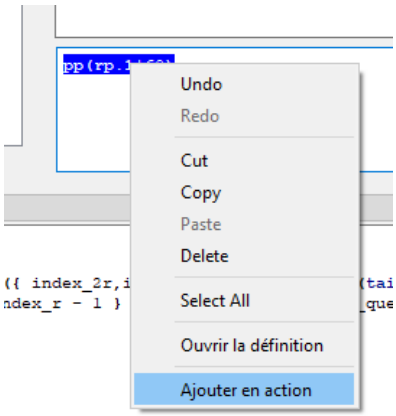
Par défaut, on utilise les capacités de raisonnement des solveurs SMT pour la logique propositionnelle et pour l'égalité. On peut également utiliser les capacités de raisonnement en logique du premier ordre (cocher «*Quantifiers*») et sur arithmétique entière (cocher «*Arithmetics*»).

La communication avec ces outils est basée sur le standard SMT-LIB, version 2.0, par un appel en ligne de commande et communications via les canaux standards. Si un outil externe a besoin de paramètres pour ce type d'interaction, alors il faut les indiquer sous «Options». Par exemple, pour l'outil Z3, les paramètres sont « -smt2 -in ».

Paramétrage de commande de preuve

Une barre d'outils configurable a été ajoutée à l'interface de preuve interactive. Elle permet aux utilisateurs d'ajouter des icônes commandant des scripts de preuves quelconques. Par exemple ci-dessous des particularisations de PP ont été ajoutées.

Ces commandes peuvent être ajoutées par le menu contextuel de l'éditeur de saisie de commandes.



[4] https://verit.loria.fr/
[5] https://github.com/Z3Prover/z3
[6] http://cvc4.cs.stanford.edu/web/

Nouvelles fonctionnalités

En plus de ces nouveautés majeures, cette version de l'Atelier B propose :

- Une nouvelle commande de preuve « ***fw(n)*** » qui permet de passer un certain nombre de commandes lors du rejeu interactif d'un script de preuve. Cette commande est aussi accessible avec le menu contextuel « *Jump To* » de l'arbre de preuve.
- L'amélioration du typage des commandes de preuve, plus efficace car s'appuyant sur le compilateur B (BComp)
- Le traducteur « *C4B* » (génération de code en langage C) permet maintenant de traduire les tableaux indexés par des énumérés.