UNIVERSITÉ DE NANTES

ECOLE DES MINES DE NANTES

cnrs

(Event-)B + probabilités : état de l'art (partiel)

Arnaud LANOIX

LABORATOIRE D'INFORMATIQUE DE NANTES ATLANTIQUE

lina

Rappel : Event-B

- Event-B machine :
  - State Variables
  - Invariant
  - Events
    - Gardes
    - Actions

```
machine AM
variables v
invariant I(v)
events
  ea ≙
  any t where
    G(t,v)
  then
    S(t,v,v')
  end
end
```

- Obligations de preuve :

- *PO(Feasibility) : $I(v) \wedge G(t,v) \vdash \exists v'.(S(t,v,v'))$*

- *PO(Invariant preservation) : $I(v) \wedge G(t,v) \wedge S(t,v,v') \vdash I(v')$*

Rappel : raffinement

- Un événement abstrait est raffiné par un (ou plusieurs) événements concrets

PO(simulation) :

I(v)

$\wedge$ J(v,w)

$\wedge$ H(u,w)

$\wedge$ T(u,w,w')

$\vdash$

$\exists$ t, v'.(G(t,v) $\wedge$ S(t,v,v') $\wedge$ J(v',w'))

machine AM
variables v
invariant I(v)
events
ea $\triangleq$
any t **where**
G(t,v)
then
S(t,v,v')
end

machine CM
refines AM
variables w
invariant J(v,w)
variant V(w)
events
ec $\triangleq$
any u **where**
H(u,w)
then
T(u,w,w')
end
end

Rappel : nouveaux événements

- Les nouveaux événements doivent **décroitre** un variant **V(w)**
  - **convergence** / terminaison des nouveaux événements

PO(variant) :

$I(v)$

$\wedge\ J(v,w)$

$\wedge\ H(u,w)$

$\vdash$

$V(w) \in \mathbb{N}$

PO(progress) :

$I(v)$

$\wedge\ J(v,w)$

$\wedge\ H(u,w)$

$\vdash$

$\forall\ w'.\ (\ T(u,w,w')\ \Rightarrow V(w') < V(w)\ )$

machine AM
variables v
invariant I(v)
events
ea ≙
any t **where**
t,v)
n
,v,v')

machine CM
variables w
invariant J(v,w)
variant V(w)
events
ec ≙
any u **where**
H(u,w)
then
T(u,w,w')
end

Rappel : non-déterminisme

- Entre événements activables (gardes G(v,t) "vraies")
- Des variables locales d'un événement
  - **any** t **where** G(t,v) **then** S(t,v,v') **end**
- Des substitutions S(t,v,v') non-déterministes :
  - ~~x := S(t,v)~~
  - x :∈ E(t,v)
  - x : ❙ Q(t,v,v')

Event-B Qualitative Probability

- *Qualitative Probabilistic Modelling in Event-B* (IFM 2007) [S. Hallerstede, T.S. Hoang]
- *Development of Rabin's Choice Coordination Algorithm in Event-B* (AVOCS'2010) [E. Yilmaz, T.S. Hoang]

- **Idée :** remplacer le choix non-déterministe par un "**choix probabiliste" : S ⊕ T**
  - *"tous les choix possibles ont une probabilité > 0 d'être considérés" (sans préciser la probabilité)*

- raisonner à propos de terminaison avec une probabilité de 1 **(almost-certain termination**)
  - événements "probabilistiquement" convergents

Exemple : partie de dés

- ***Enoncé*** *:*
  - *on lance 2 dés à 6 faces ;*
  - *la partie est gagnée si les deux dés sont égaux*

- On a un "**choix probabiliste"** pour la valeur d'un dé,
  - i.e. toutes les valeurs ont la même probabilité d'arriver

- Un nombre indéterminé de lancers doit permettre d'aboutir

Qualitative Probabilistic choice: proposition de [Hallerstede & al]

- Une nouvelle notation : **x ⊕ ❙ Q(t,v,v')**
  - au lieu de x : ❙ Q(t,v,v')

- Des événements non-déterministes + des événements probabilistes :
  - pas d'événements mixant des choix déterministes + choix probabilistes

- Nouvelles obligations de preuves définies

- Ajout d'une "borne maximale" **U(w)** pour "borner" le variant

- Le choix probabiliste **raffine** le choix non-déterministe

Probabilistic events: obligations de preuves

- *PO(Invariant préservation)* : identique
- *PO(Progress)*
  - Non-deterministic action :
    - "**must** decrease the variant"
  - Probabilistic action :
    - "**may** decrease the variant"
- *PO (feasiblity)* déduite

machine AM
variables v
invariant I(v)
events
ea ≙
any t **where**
G(t,v)
then
S(t,v,v')
end

machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
ec ≙
any u **where**
H(u,w)
then
T(u,w,w')
end

Probabilistic events: obligations de preuves

machine AM
variables v
invariant I(v)
events
ea ≙
any t **where**
G(t,v)
then
S(t,v,v')
end

machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
ec ≙
any u **where**
H(u,w)
then
T(u,w,w')
end

- *PO(Invariant préservation)* : identique
- *PO(Progress)*
  - Non-deterministic action :
    - "**must** decrease the variant"
  - Probabilistic action :
    - "**may** decrease the variant"
- *PO (feasiblity)* déduite

PO(progress) :

$I(v)$

$\wedge\ J(v,w)$

$\wedge\ H(u,w)$

$\vdash$

$\forall\ w'.\ (\ T(u,w,w')\ \Rightarrow V(w') < V(w)\ )$

Probabilistic events: obligations de preuves

machine AM
variables v
invariant I(v)
events
ea ≙
any t **where**
G(t,v)
then
S(t,v,v')
end

machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
ec ≙
any u **where**
H(u,w)
then
T(u,w,w')
end

- *PO(Invariant préservation)* : identique
- *PO(Progress)*
  - Non-deterministic action :
    - "**must** decrease the variant"
  - Probabilistic action :
    - "**may** decrease the variant"
- *PO (feasiblity)* déduite

PO(progress) :

I(v)

∧ J(v,w)

PO(almost-certain progress) :

I(v)

∧ J(v,w)

∧ H(u,w)

⊢

∃ w'. (T(u,w,w') ∧ V(w') < V(w))

Probabilistic events: nouvelles OPs

- *PO(upperBound)*
- *PO(almost-certain convergence)*
- *PO(finiteness)*

```
machine AM
variables v
invariant I(v)
events
  ea ≙
  any t where
    G(t,v)
  then
    S(t,v,v')
  end
```

```
machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
  ec ≙
  any u where
    H(u,w)
  then
    T(u,w,w')
  end
end
```

Probabilistic events: nouvelles OPs

- *PO(upperBound)*

$$I(v) \wedge J(v,w) \wedge H(u,w) \vdash V(w) \leq U(w)$$

- *PO(almost-certain convergence)*

- *PO(finiteness)*

```
machine AM
variables v
invariant I(v)
events
  ea ≙
  any t where
    G(t,v)
  then
    S(t,v,v')
  end
```

```
machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
  ec ≙
  any u where
    H(u,w)
  then
    T(u,w,w')
  end
end
```

Probabilistic events: nouvelles OPs

- *PO(upperBound)*

$$I(v) \wedge J(v,w) \wedge H(u,w) \vdash V(w) \leq U(w)$$

- *PO(almost-certain convergence)*

$$I(v) \wedge J(v,w) \wedge H(u,w) \wedge T(u,w,w') \vdash U(w') \leq U(w)$$

- *PO(finiteness)*

```
machine AM
variables v
invariant I(v)
events
  ea ≙
  any t where
    G(t,v)
  then
    S(t,v,v')
  end
```

```
machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
  ec ≙
  any u where
    H(u,w)
  then
    T(u,w,w')
  end
end
```

Probabilistic events: nouvelles OPs

- *PO(upperBound)*

$$I(v) \wedge J(v,w) \wedge H(u,w) \vdash V(w) \leq U(w)$$

- *PO(almost-certain convergence)*

$$I(v) \wedge J(v,w) \wedge H(u,w) \wedge T(u,w,w') \vdash U(w') \leq U(w)$$

- *PO(finiteness)*

$$I(v) \wedge J(v,w) \wedge H(u,w) \vdash \text{finite}(\{ w' \mid T(u,w,w') \})$$

```
machine AM
variables v
invariant I(v)
events
  ea ≙
  any t where
    G(t,v)
  then
    S(t,v,v')
  end
```

```
machine CM
variables w
invariant J(v,w)
variant V(w)
bound U(w)
events
  ec ≙
  any u where
    H(u,w)
  then
    T(u,w,w')
  end
end
```

Un plugin pour Rodin

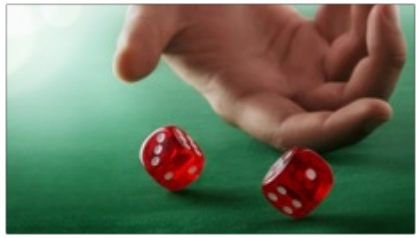
- http://wiki.event-b.org/index.php/Event-B_Qualitative_Probability_User_Guide

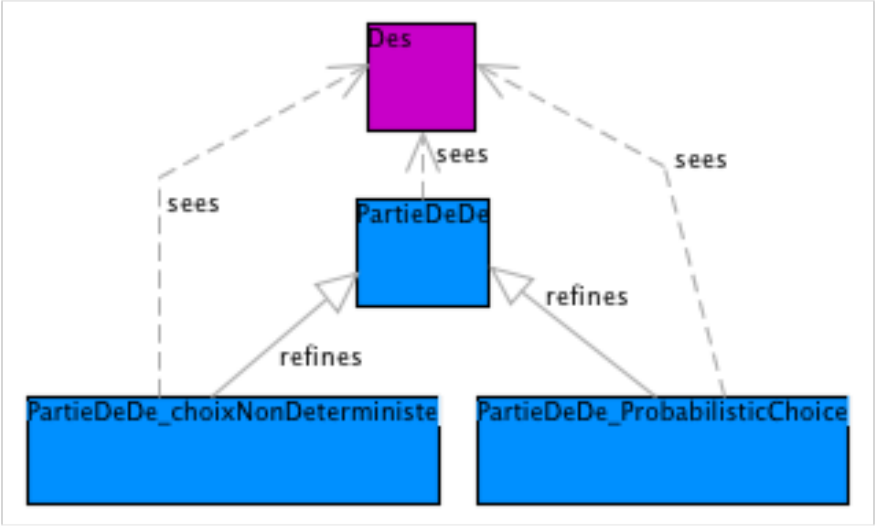
- **Extensions** du langage Event-B
  - event qualifier : standard / probabilistic
    - ~~au lieu de x ⊕ **|** Q(t,v,v')~~
  - bound element

- Implémentation des **Obligations de Preuves** nécessaires

- *Dernière MAJ du plugin : 14.05.2012*

Modélisation Event-B





La solution "en un coup"

Plusieurs lancers,
jusqu'à aboutir

- **TODO :** prouver la **convergence** des lancers afin de montrer que l'on abouti bien à une solution

```
CONTEXT
    Des      ›
CONSTANTS
o   FACES    ›
o   diff     ›
AXIOMS
o   axm1:   FACES = 1‥6 not theorem ›Valeurs possible pour un dé
o   axm2:   diff ∈ (FACES × FACES) ⟶ 0‥1 not theorem ›
o   axm3:   ∀ x,y·(x∈ FACES ∧ y∈ FACES ∧ x↦y ∈ dom(diff) ∧ x≠y ⇒ diff(x↦y) = 1) not theorem ›
o   axm4:   ∀ x,y·(x∈ FACES ∧ y∈ FACES ∧ x↦y ∈ dom(diff) ∧ x=y ⇒ diff(x↦y) = 0) not theorem ›
END
```

```
CONTEXT
    Des      ›
CONSTANTS
o   FACES    ›
o   diff     ›
AXIOMS
o   axm1:   FACES = 1‥6 not theorem ›Valeurs possible pour un dé
o   axm2:   diff ∈ (FACES × FACES) ⟶ 0‥1 not theorem ›
o   axm3:   ∀ x,y·(x∈ FACES ∧ y∈ FACES ∧ x↦y ∈ dom(diff) ∧ x≠y ⇒ diff(x↦y) = 1) not theorem ›
o   axm4:   ∀ x,y·(x∈ FACES ∧ y∈ FACES ∧ x↦y ∈ dom(diff) ∧ x=y ⇒ diff(x↦y) = 0) not theorem ›
END
```

```
MACHINE
    PartieDeDe  ›
SEES
o    Des
VARIABLES
o   de1Gagnant    ›
o   de2Gagnant    ›
INVARIANTS
o   inv1:   de1Gagnant ∈ FACES
o   inv2:   de2Gagnant ∈ FACES
```

```
o   LancerGagnant:   not extended ordinary standard ›Solution du problème "en un coup"
    WHERE
    o   grd1:   de1Gagnant ≠ de2Gagnant not theorem ›
    THEN
    o   act1:   de1Gagnant, de2Gagnant :| de1Gagnant' ∈ FACES ∧ de2Gagnant' ∈ FACES ∧ de1Gagnant' = de2Gagnant' ›
    END
```

```
MACHINE
    PartieDeDe_choixNonDeterministe
REFINES
    PartieDeDe
SEES
    Des
VARIABLES
    de1 ∈
    de2 ∈
    de1Gagnant ∈
    de2Gagnant ∈
INVARIANTS
    inv1:   de1 ∈ FACES not theorem ∈
    inv2:   de2 ∈ FACES not theorem ∈
VARIANT
    diff(de1↦de2) ∈
```

```
MACHINE
    PartieDeDe_choixNonDeterministe
REFINES
  o    PartieDeDe
SEES
  o    Des
VARIABLES
  o   de1 ;
  o   de2 ›
  o   de1Gagnant    ›
  o   de2Gagnant    ›
INVARIANTS
  o   inv1:   de1 ∈ FACES not theorem ›
  o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
  o   diff(de1↦de2) ›
```

```
lancer:  not extended convergent  ›
WHERE
  o   grd1:   de1 ≠ de2 not theorem ›
THEN
  o   act1:   de1 :∈ FACES ›
  o   act2:   de2 :∈ FACES ›
END
```

```
MACHINE
    PartieDeDe_choixNonDeterministe
REFINES
  o   PartieDeDe
SEES
  o   Des
VARIABLES
  o   de1 ›
  o   de2 ›
  o   de1Gagnant ›
  o   de2Gagnant ›
INVARIANTS
  o   inv1:   de1 ∈ FACES not theorem ›
  o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
  o   diff(de1↦de2) ›
```

```
lancer:  not extended convergent ›
WHERE
  o   grd1:   de1 ≠ de2 not theorem ›
THEN
  o   act1:   de1 :∈ FACES ›
  o   act2:   de2 :∈ FACES ›
END
```

```
PartieGagnee:    not extended ordinary ›
REFINES
  o   LancerGagnant
WHERE
  o   grd1:   de1Gagnant ≠ de2Gagnant not theorem ›
  o   grd2:   de1 = de2 not theorem ›
THEN
  o   act2:   de1Gagnant, de2Gagnant = de1, de2 ›
END
```

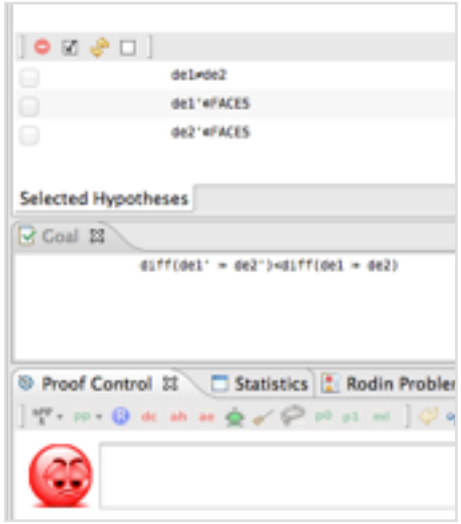
```
MACHINE
    PartieDeDe_choixNonDeterministe
REFINES
    PartieDeDe
SEES
    Des
VARIABLES
    de1
    de2
    de1Gagnant
    de2Gagnant
INVARIANTS
    inv1:   de1 ∈ FACES not theorem
    inv2:   de2 ∈ FACES not theorem
VARIANT
    diff(de1↦de2)
```

```
lancer:  not extended convergent
WHERE
    grd1:   de1 ≠ de2 not theorem
THEN
    act1:   de1 :∈ FACES
    act2:   de2 :∈ FACES
END
```

Impossible de **prouver** la convergence
i.e. la décroissance du variant
OP(progress) ???

Il faudrait que cela soit vrai pour toutes les valeurs possibles de de1 et de2

```
PartieGagnee:    not extended ordinary
REFINES
    LancerGagnant
WHERE
    grd1:   de1Gagnant ≠ de2Gagnant not theorem
    grd2:   de1 = de2 not theorem
THEN
    act2:   de1Gagnant, de2Gagnant = de1, de2
END
```



```
MACHINE
    PartieDeDe_ProbabilisticChoice ›
REFINES
o    PartieDeDe
SEES
o    Des
VARIABLES
o   de1Gagnant   ›
o   de2Gagnant   ›
o   de1  ›
o   de2  ›
INVARIANTS
o   inv1:   de1 ∈ FACES not theorem ›
o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
o   diff(de1↦de2) ›
BOUND
o   1 ›
```

```
o   lancer:  not extended convergent probabilistic ›
    WHERE
    o   grd1:   de1 ≠ de2 not theorem ›
    THEN
    o   act1:   de1 :∈ FACES ›
    o   act2:   de2 :∈ FACES ›
    END
```

```
MACHINE
    PartieDeDe_ProbabilisticChoice ›
REFINES
o    PartieDeDe
SEES
o    Des
VARIABLES
o   de1Gagnant   ›
o   de2Gagnant   ›
o   de1  ›
o   de2  ›
INVARIANTS
o   inv1:   de1 ∈ FACES not theorem ›
o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
o   diff(de1↦de2) ›
BOUND
o   1 ›
```

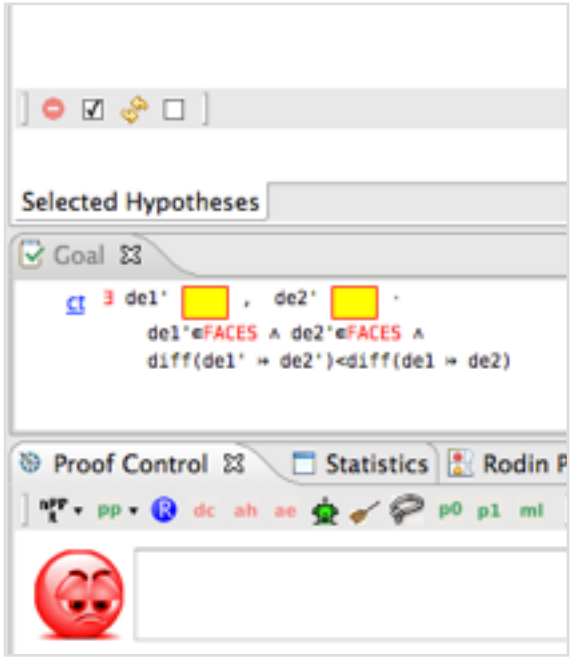
```
o   lancer:  not extended convergent probabilistic ›
    WHERE
    o   grd1:   de1 ≠ de2 not theorem ›
    THEN
    o   act1:   de1 :∈ FACES ›
    o   act2:   de2 :∈ FACES ›
    END
```

```
MACHINE
    PartieDeDe_ProbabilisticChoice  ›
REFINES
o    PartieDeDe
SEES
o    Des
VARIABLES
o   de1Gagnant    ›
o   de2Gagnant    ›
o   de1   ›
o   de2   ›
INVARIANTS
o   inv1:   de1 ∈ FACES not theorem ›
o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
o   diff(de1↦de2) ›
BOUND
o   1 ›
```

```
o   lancer:  not extended convergent probabilistic ›
    WHERE
    o   grd1:   de1 ≠ de2 not theorem ›
    THEN
    o   act1:   de1 :∈ FACES ›
    o   act2:   de2 :∈ FACES ›
    END
```

on peut prouver le "*almost-certain progress*" de l'événement lancer

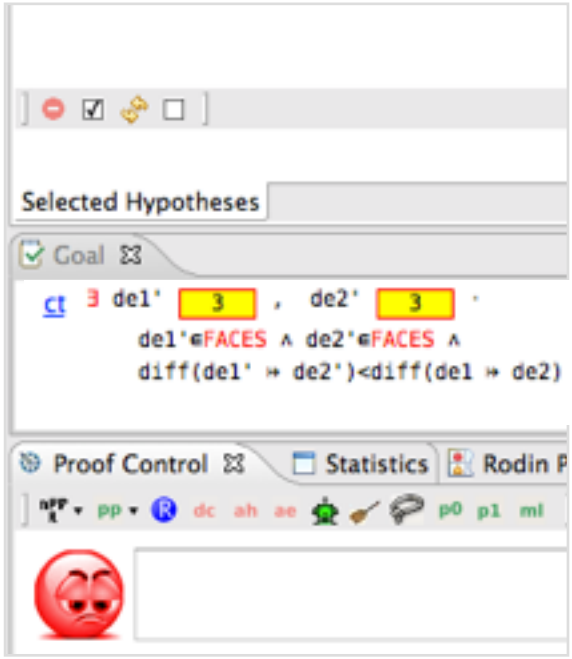
Il s'agit uniquement de montrer qu'il existe des valeurs possibles pour de1 et de2 telles que le variant décroit



```
MACHINE
    PartieDeDe_ProbabilisticChoice ›
REFINES
o    PartieDeDe
SEES
o    Des
VARIABLES
o   de1Gagnant   ›
o   de2Gagnant   ›
o   de1   ›
o   de2   ›
INVARIANTS
o   inv1:   de1 ∈ FACES not theorem ›
o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
o   diff(de1↦de2) ›
BOUND
o   1 ›
```

```
o   lancer:  not extended convergent probabilistic ›
    WHERE
    o   grd1:   de1 ≠ de2 not theorem ›
    THEN
    o   act1:   de1 :∈ FACES ›
    o   act2:   de2 :∈ FACES ›
    END
```

on peut prouver le "*almost-certain progress*" de l'événement lancer

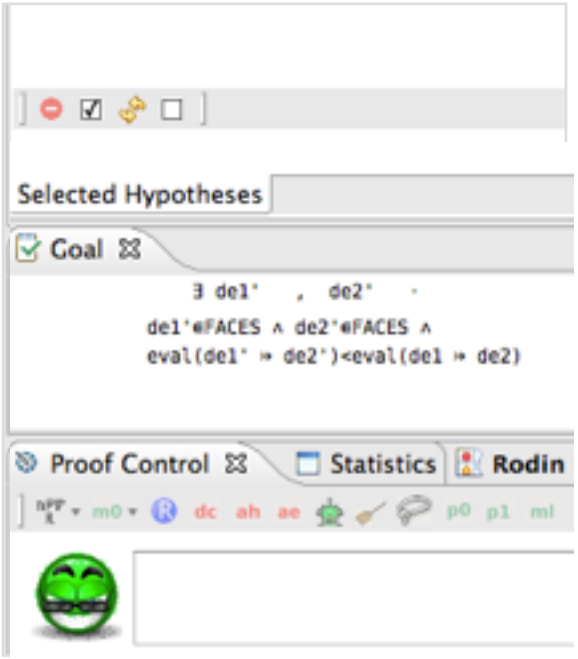


Il s'agit uniquement de montrer qu'il existe des valeurs possibles pour de1 et de2 telles que le variant décroit

```
MACHINE
    PartieDeDe_ProbabilisticChoice ›
REFINES
o    PartieDeDe
SEES
o    Des
VARIABLES
o   de1Gagnant   ›
o   de2Gagnant   ›
o   de1   ›
o   de2   ›
INVARIANTS
o   inv1:   de1 ∈ FACES not theorem ›
o   inv2:   de2 ∈ FACES not theorem ›
VARIANT
o   diff(de1↦de2) ›
BOUND
o   1 ›
```

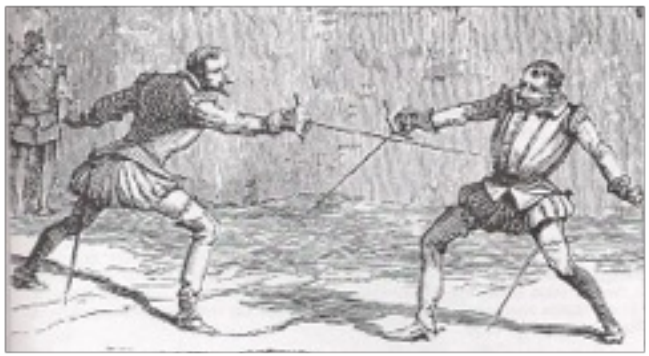
```
o   lancer:  not extended convergent probabilistic ›
    WHERE
    o   grd1:   de1 ≠ de2 not theorem ›
    THEN
    o   act1:   de1 :∈ FACES ›
    o   act2:   de2 :∈ FACES ›
    END
```

on peut prouver le "*almost-certain progress*" de l'événement lancer



Il s'agit uniquement de montrer qu'il existe des valeurs possibles pour de1 et de2 telles que le variant décroit

Exemple : mousquetaires



```
MACHINE
    Duel    ›
SEES
o    MousquetaireContext
VARIABLES
o   duelistes    ›
o   vainqueur    ›
INVARIANTS
o   inv1:   duelistes ⊆ DUELISTES not
o   inv2:   duelistes ≠ ∅ not theorem
o   inv3:   vainqueur ∈ DUELISTES not
VARIANT
o   duelistes ›
BOUND
o   DUELISTES ›
EVENTS
o   INITIALISATION:  not extended ordi
    THEN
    o   act1:   duelistes = DUELISTES
    o   act2:   vainqueur :∈ DUELISTES
    END
```

```
DArtaganAttaque:     not extended convergent probabilistic ›
WHERE
o   grd1:   duelistes = DUELISTES not theorem ›
THEN
o   act1:   duelistes :| duelistes' = DUELISTES ∨  duelistes' = {DArtagnan}
END

RochefortAttaque:    not extended convergent probabilistic ›
WHERE
o   grd1:   duelistes = DUELISTES not theorem ›
THEN
o   act1:   duelistes :| duelistes' = DUELISTES ∨  duelistes' = {Rochefort}
END

finDuDuel:    not extended ordinary   ›
WHERE
o   grd1:   card(duelistes) = 1 not theorem ›
THEN
o   act1:   vainqueur :∈ duelistes ›
END
```

Autres références bibliographiques

Probabilistic termination in B (ZB'2003)
[A. McIver, C. Morgan, T.S. Hoang]

Probabilistic invariants for probabilistic machines. (ZB'2003)
[T.S. Hoang, Z. Jin, K. Robinson, A. McIver, C. Morgan]

The challenge of probabilistic Event B (ZB'05)
[C. Morgan, T.S. Hoang, J.-R. Abrial]

Development via refinement in probabilistic b: fondation and case study (ZB'05) [T.S. Hoang, Z. Jin, K. Robinson, A. McIver, C. Morgan]

The Development of a Probabilistic B-Method and Supporting Toolkit
PhD Thesis (july 2005) [T.S. Hoang]

Tool support for qualitative reasoning in Event-B
Master thesis (Aug. 2010) [E. Yilmaz]

Quantitative probability in Event-B

Towards Probabilistic Modelling in Event-B (IFM 2010) [A. Tarasyuk, E. Troubitsyna, L. Laibinis]

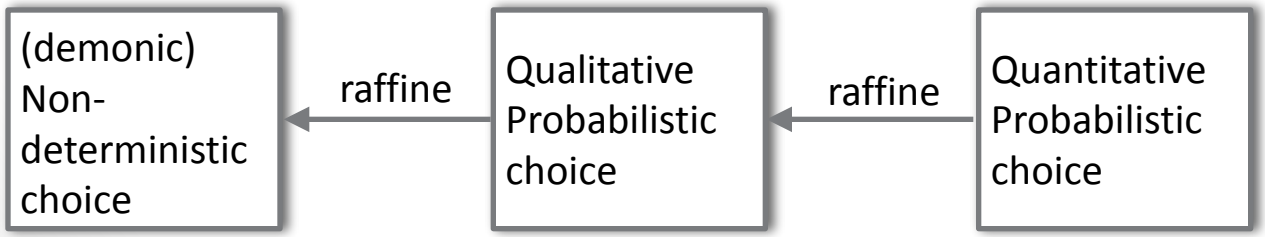
Quantitative verification of system safety in Event-B (SERENE 2011) [A. Tarasyuk, E. Troubitsyna, L. Laibinis]

Formal modelling and verification of service-oriented systems in probabilistic Event-B (IFM 2012) [A. Tarasyuk, E. Troubitsyna, L. Laibinis]

- **Idée :** quantifier les différentes probabilités
  - nouvel opérateur de **choix probabiliste quantifié** :

$$x \oplus | \; x_1 @ p_1 ; x_2 @ p_2 ; \cdots ; x_n @ p_n \quad \text{avec} \quad \sum_{i=1}^{n} p_i = 1$$

$$\forall p_i \,.\, p_i > 0$$



Quantitative probability in Event-B

- Articles beaucoup moins "techniques" / moins "pratiques"

Quantitative verification of system safety in Event-B (SERENE 2011) [A. Tarasyuk, E. Troubitsyna, L. Laibinis]

- **méthodologie** pour quantifier la probabilité de "casser" une propriété de sûreté exprimées dans l'invariant
  - Des probabilités $\mathbf{p_1, p_2, p_3}$ que des événements produisent un échec sont introduites par raffinement ;
  - On en déduit une **probabilité "globale"**
    - *exemple:* $\mathbf{P_{SAF} = (1-p_1).(1-p_2)^2 + (1-p_3).(1 - (1-p_1)^2.p_3)}$
      - *idée : "raisonner" à partir des OPs (preservation inv)*
  - On peut ensuite **instancier** p_1, p_2, p_3...

Formal modelling and verification of service-oriented systems in probabilistic Event-B (IFM 2012) [A. Tarasyuk, E. Troubitsyna, L. Laibinis]

- Utilisation de **CTMC** (continuous-time Markov chain)