Extensive Infinite Games and Escalation, an exercice in Agda

Pierre Lescanne

Extensive Infinite Games and Escalation, an exercice in Agda

Pierre Lescanne

University of Lyon, École normale supérieure de Lyon, CNRS (LIP),
46 allée d'Italie, 69364 Lyon, France

Escalation in games is when agents keep playing forever. Based on formal proofs we claim that *if agents assume that resource are infinite, escalation is rational.*

Keywords: extensive game, infinite game, sequential game, escalation coinduction, Agda, proof assistant, formal proof.

1 Introduction

Escalation in games is the phenomenon where agents keep playing (or betting if the game consists in bets) forever, leading to their ruin. Since Shubik[12] people claim that such an attitude is not rational. Based on formal proofs we are able to refute such a claim and to say that *if agents assume that resource are infinite, escalation is rational.* Since our first work[4] which took place before the 2008 financial crisis, evidence[1] show that stating the rationality of escalation makes sense. The only solution for avoiding escalation is then to assume that resource are finite.

In previous works[6,5] we used an approach based on Coq[2] and coinduction (a dual of induction aimed at reasoning on infinite data structures[3]). Especially in[7] we used dependent types together with coinduction. In this paper, we use coinduction in Agda[9], because it allows a terse style closed to this of mathematicians. Agda is a formal proof computer environment as well as a dependently typed programming language.

Notice other works using proof assistants for proving properties of agents. For instance, Stéphane Le Roux proved the existence of Nash equilibria using Coq and Isabelle[10,11]. In a somewhat connected area, Tobias Nipkow proved Arrows theorem in HOL[8]. Agda code of this development are available on GitHub [1].

2 Games and Strategy Profiles

Since we study game theory, lest us first define games. A *game* is either a *leaf* or a *node*. A leaf is a assignment to each agent of a *Utility* (sometime called a *payoff*). Note that the type of utility depends on the agent (dependent type). A node contains two entities, put in a record: an *agent* (the agent who has the trait) and a function *next* which tells the next positions to be played.

[1] `https://github.com/PierreLescanne/DependentTypesForExtensiveGames-in-Agda`

```
mutual
  Game = ((a : Agent) → Utility a) ⊎ NodeG

  record NodeG : Set where
    coinductive
    field
      ag : Agent
      next : Choice → Game
```

Notice the key word coinductive which shows that we deal with infinite games. The main concept in game theory is this of *strategy profiles*. Strategy profiles are like games with at each node a choice, which is the choice of the agent who continues the game. In Agda the sum comes with to unctions inj_1 and inj_2. In our case, if u is a utility assignment of type ((a : Agent) → Utility a) then inj_1 u is a Game and n is a NodeG then inj_2 n is a Game. Strategy profiles are abbreviated StratProf.

```
mutual
  StratProf = ((a : Agent) → Utility a) ⊎ NodeS

  record NodeS : Set where
    coinductive
    field
      ag : Agent
      next : Choice → StratProf
      ch : Choice
```

We can define the underlying game of a strategy profile

```
game : (s : StratProf) → Game
game (inj₁ u) = inj₁ u
game (inj₂ n) = inj₂ (gameN n) where
  gameN : NodeS → NodeG
  NodeG.ag (gameN n) = ag n
  NodeG.next (gameN n) c = game (next n c)
```

The underlying game of a leaf (strategy profile) is the same utility assignment, i.e., a leaf (game). For nodes, games are attributed corecursively. Now let us look at another concept. Given two strategy profiles, one may wonder whether they have the same underlying game. This is given by the binary relation $\_\approx^{sg}\_$.

```
mutual
  data _≈ˢᵍ_ : StratProf → StratProf → Set where
    ≈ˢᵍLeaf : u : (a : Agent) → Utility a → inj₁ u ≈ˢᵍ inj₁ u
    ≈ˢᵍNode : n n' : NodeS → n ○≈ˢᵍ n' → inj₂ n ≈ˢᵍ inj₂ n'

  record _○≈ˢᵍ_ (n n' : NodeS) : Set where
    coinductive
    field
      is○≈ˢᵍ : ag n ≡ ag n' → ((c : Choice) → next n c ≈ˢᵍ next n' c)
```

A leaf has the same game as itself, two nodes have the same game if all their "next" strategy profiles have the same games. Notice that we use the symbol ∘ for concepts associated with NodeS, when the concept without ∘ is associated with StratProf. Given a strategy profile, we may want to compute the utility of an agent. This assumes that the path that follows the choices of the agents leads to a leaf. A strategy profile s with such a property is said convergent, written ↓ s. This is defined as follows:

```
mutual
  data ↓ : StratProf → Set where
    ↓Leaf : u : (a : Agent) → Utility a → ↓ (inj₁ u)
    ↓Node : n : NodeS → ∘↓ n → ↓ (inj₂ n)

  record ∘↓ (n : NodeS) : Set where
    inductive
    field
      is∘↓ : ↓ (next n (ch n))
```

Notice that not all the strategy profile are convergent, for instance the strategy profile AcBc of Section 4 is not convergent.

We define the utility assignment ⓤ of a convergent strategy profile. ⓤ takes two parameters: a strategy profile s and a proof that s is convergent.

```
  ⓤ : (s : StratProf) → (↓ s) → (a : Agent) → Utility a
  ⓤ (inj₁ u) ↓Leaf = u
  ⓤ (inj₂ n) (↓Node p) = ∘ⓤ n p

  ∘ⓤ : (n : NodeS) → (∘↓ n)  → (a : Agent) → Utility a
  ∘ⓤ n p = ⓤ (next n (ch n)) (is∘↓ p)
```

Subgame perfect equilibria are very interesting strategy profiles. They are strategy profiles in which the choices of the agents are the best. A leaf is always a subgame perfect equilibrium. A node is a subgame perfect equilibrium if the next strategy profile for the choice of the agent is convergent and is a subgame perfect equilibrium, if for any other node which has the same game and whose next strategy profile is also convergent and is a subgame perfect equilibrium, the utility of the agent of the given node is not less than the utility of the agent of this other node. This is defined formally in Agda as follows, where we use ⥫ s to tell that s is a subgame perfect equilibrium.

```
data ⥫_ : StratProf → Set where
  ⥫Leaf : u : (a : Agent) → Utility a → ⥫ inj₁ u
  ⥫Node : n n' : NodeS →
    n ∘≈^sg n' →
     ⥫ (next n (ch n)) →
     ⥫ (next n' (ch n')) →
    (p : ↓ (next n (ch n))) → (p' : ↓ (next n' (ch n'))) →
    (ⓤ (next n (ch n)) p (ag n)) ≮ (ⓤ (next n' (ch n')) p' (ag n)) →
     ⥫ inj₂ n
```

3 Escalation

We are now interested in strategy profile leading to escalation.

3.1 Good strategy profile

A first property toward escalation is what we call *goodness*. A strategy profile is *good* if at each node, there is a subgame perfect equilibrium with the same game and the same choice.

```
mutual
  data ☺_ : (s : StratProf) → Set where
    ☺Node :  n : NodeS → ∘☺ n → ☺ (inj₂ n)
  record ∘☺_ (n : NodeS) : Set where
    coinductive
    field
      is∘☺ : (n' : NodeS) → ⇌ (inj₂ n') → n ∘≈^sg n' → ch n ≡ ch n' →
        ☺ (next n (ch n))
```

In other words, this strategy profile is not itself a subgame perfect equilibrium, in particular, it can be non convergent, but each of its choices is dictated by a subgame perfect equilibrium. Goodness can be considered as *rationality* in the choices of the agents. Reader may notice that goodness is of interest only in infinite games, because in a finite game, there is no difference between a good strategy and a subgame perfect equilibrium.

3.2 Divergent strategy profile

Another property of strategy profiles is *divergence*. In a divergent strategy profile, if one follows the choices of the agents, one never gets to a leaf, but, on the opposite, one runs forever. A divergent strategy profile is written ↑ s. The formal definition in Agda of divergence looks like this of convergence, but the test for divergence is based on a coinductive record and never hits a leaf, therefore there is no ↑Leaf case.

```
mutual
  data ↑_ : StratProf → Set where
    ↑Node :  n : NodeS → ∘↑ n → ↑ (inj₂ n)

  record ∘↑ (n : NodeS) : Set where
    coinductive
    field
      is∘↑ : ↑ (next n (ch n))
```

An *escalation* is a strategy profile which is both *good* and *divergent*.

4 Strategies with two agents and two choices

To build escalating strategy profiles, we consider the case of two agents Alice and Bob and two choices down and right.

```
data AliceBob : Set where
  Alice Bob : AliceBob
```

```
data DorR : Set where
  down right : DorR
```

We take the natural numbers $\mathbb{N}$ as utility[2] for both agents[3] and for the $\not<$ relation we take the $\succcurlyeq$ relation defined as:

```
data _≽_ : ℕ → ℕ → Set where
  z≽z : zero ≽ zero
  s≽z : n : ℕ → suc n ≽ zero
  s≽s : n m : ℕ → n ≽ m → suc n ≽ suc m
```

A utility assignment is for instance this which assigns 1 to Alice and 0 to Bob:

```
uA1B0 : AliceBob → ℕ
uA1B0 Alice = 1
uA1B0 Bob = 0
```

from which we can build a leaf strategy profile:

```
A1B0 : StratProf
A1B0 = inj₁ uA1B0
```

which is convergent.

```
↓A1B0 : ↓ A1B0
↓A1B0 = ↓Leaf
```

From the utility assignment which assigns 0 to Alice and 1 to Bob on can build the convergent strategy profile `A0B1`.

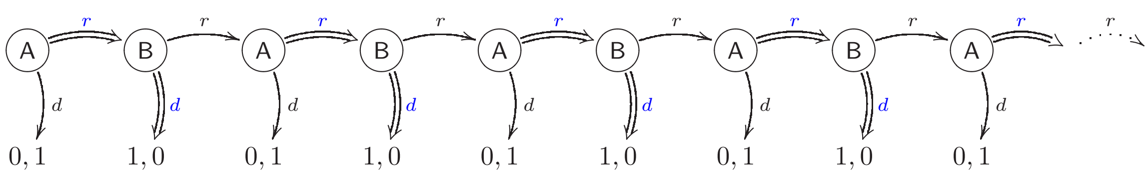
Moreover, we build an infinite strategy AcBs, in which Alice continues always and Bob stops always:

[2] We could have taken a utility with only two values, but we feel that the reader is more acquainted with natural numbers for utilities.

[3] In this case, the type of utility does not depend on the agent.

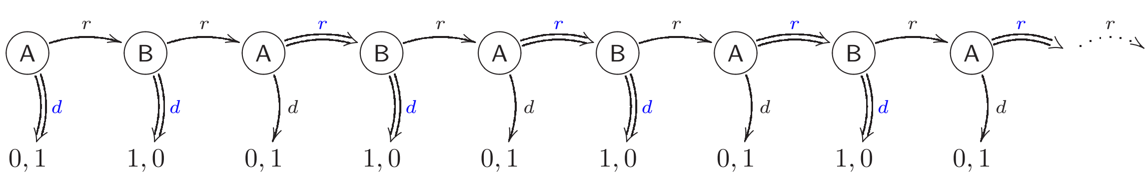
```
mutual
  AcBs : StratProf
  AcBs = inj₂ ∘AcBs

  ∘AcBs : NodeS
  ag ∘AcBs  = Alice
  ch ∘AcBs  = right
  next ∘AcBs down = A0B1
  next ∘AcBs right =  BsAc

  BsAc : StratProf
  BsAc = inj₂ ∘BsAc

  ∘BsAc : NodeS
  ag ∘BsAc = Bob
  ch ∘BsAc = down
  next ∘BsAc down = A1B0
  next ∘BsAc right = AcBs
```

We notices that by mutual co-recursion, AcBs is defined together with an infinite strategy profile BsAc which starts with a node of which Bob is the agent. Those strategies are like infinite combs.



With down one reaches always a leaf and with right one goes always to a new strategy profile, which is a node. There is a variant of the node ∘AcBs, in which the first choice of Alice is down instead of right.

```
Var∘AcBs : NodeS
ag Var∘AcBs  = Alice
ch Var∘AcBs  = down
next Var∘AcBs down = A0B1
next Var∘AcBs right = BsAc
```



We prove that ∘AcBs and Var∘AcBs have the same game. Likewise we prove that AcBs is convergent i.e., $\downarrow$ AcBs. Those two facts are key steps in the proof that AcBs is subgame prefect equilibrium i.e., that $\rightleftharpoons$ AcBs.

On the same paradigm we built a strategy profile AsBc in which A stops and B continues and which is proved to be convergent and to be a subgame perfect equilibrium.

We also build a strategy profile in which A and B both continue.

```
mutual
  AcBc : StratProf
  AcBc = inj₂ ∘AcBc

  ∘AcBc : NodeS
  ag ∘AcBc  = Alice
  ch ∘AcBc  = right
  next ∘AcBc down = A0B1
  next ∘AcBc right =  BcAc

  BcAc : StratProf
  BcAc = inj₂  ∘BcAc

  ∘BcAc : NodeS
  ag ∘BcAc = Bob
  ch ∘BcAc = right
  next ∘BcAc down = A1B0
  next ∘BcAc right = AcBc
```

AcBs, AcBc and AsBc have the same game. Unlike AcBs and AsBc, the strategy profile AcBc is divergent, i.e., ↑AcBc. Moreover AcBc is good which means ☺AcBc.

5 Conclusion

Since AcBc is good and divergent, AcBc **is an escalation**. Hence we proved formally the claim of the introduction, namely *if agents assume that resource are infinite, escalation is rational*.

In the current implementation, the type of choices is the same for all the agents. However, one may imagine that this type may depend on the agents. Making the type of choices depending on the agents is object of the current investigation.

References

1. Lucie M Bland, Jessica A Rowland, Tracey J Regan, David A Keith, Nicholas J Murray, Rebecca E Lester, Matt Linn, Jon Paul Rodríguez, and Emily Nicholson. Developing a standardized definition of ecosystem collapse for risk assessment. *Frontiers in Ecology and the Environment*, 16(1):29–36, 2018.
2. Pierre Boutillier, Stephane Glondu, Benjamin Grégoire, Hugo Herbelin, Pierre Letouzey, Pierre-Marie Pédrot, Yann Régis-Gianas, Matthieu Sozeau, Arnaud Spiwack, and Enrico Tassi. Coq 8.4 Reference Manual. Research report, Inria, July 2014. The Coq Development Team.
3. Bart Jacobs and Jan Rutten. An introduction to (co)algebra and (co)induction. In Davide Sangiorgi and Jan J. M. M. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge tracts in theoretical computer science*, pages 38–99. Cambridge University Press, 2012.

4. Pierre Lescanne. (Mechanical) Reasoning on Infinite Extensive Games. *CoRR*, abs/0805.1798, 2008. avaliable on `http://arxiv.org/abs/0805.1798`.
5. Pierre Lescanne. Rationality and escalation in infinite extensive games. *CoRR*, abs/1112.1185, 2011. available on `http://arxiv.org/abs/1112.1185`.
6. Pierre Lescanne. Bubbles are rational. *CoRR*, abs/1305.0101, 2013.
7. Pierre Lescanne. Dependent types for extensive games. *J. Formalized Reasoning*, 11(1):1–17, 2018. available at `https://jfr.unibo.it/article/view/7517`.
8. Tobias Nipkow. Social choice theory in HOL. *J. Autom. Reasoning*, 43(3):289–304, 2009. Available at `http://www21.in.tum.de/~nipkow/pubs/arrow.pdf`.
9. Ulf Norell. Dependently typed programming in Agda. In Andrew Kennedy and Amal Ahmed, editors, *Proceedings of TLDI'09: 2009 ACM SIGPLAN International Workshop on Types in Languages Design and Implementation, Savannah, GA, USA, January 24, 2009*, pages 1–2. ACM, 2009.
10. Stéphane Le Roux. Acyclic preferences and existence of sequential Nash equilibria: A formal and constructive equivalence. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *Theorem Proving in Higher Order Logics, 22nd International Conference, TPHOLs 2009, Munich, Germany, August 17-20, 2009. Proceedings*, volume 5674 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2009. available at `http://www.ens-lyon.fr/LIP/Pub/Rapports/RR/RR2007/RR2007-18.pdf`.
11. Stephane Le Roux, Erik Martin-Dorel, and Jan-Georg Smaus. Formalization of an existence theorem of Nash equilibrium in Coq and Isabelle. In Patricia Bouyer and Pierluigi San Pietro, editors, *GandALF 2017, The Eighth International Symposium on Games, Automata, Logics, and Formal Verification*. EPTCS 256, September 2017. Available at `http://eptcs.web.cse.unsw.edu.au/content.cgi?GANDALF2017`.
12. Martin Shubik. The dollar auction game: A paradox in noncooperative behavior and escalation. *Journal of Conflict Resolution*, 15(1):109–111, 1971.