Lecture 8: Argumentation II

Peter McBurney

(with thanks to Elizabeth Black)

Today

1. Introduction to Artificial Intelligence (FMT)
2. Probabilistic Reasoning I (FMT)
3. Probabilistic Reasoning II (FMT)
4. Sequential Decision Making (FMT)
5. Game Theory (FMT)
6. Temporal Probabilistic Reasoning (FMT)
7. Argumentation I (PMcB)
8. Argumentation II (PMcB)
9. (A peek at) Machine Learning (PMcB)
10. AI & Ethics (SS)

Today

- Introduction
- Methods for computing semantics
  - Equational approach for determining complete extensions
  - Argument game approach for determining membership of grounded extension
  - Decomposition approach for determining extensions

Introduction - recap

Briefly considered different types of argument (adversarial/cooperative, dialogical/statement, internal/multi-party).

Abstract argumentation frameworks.

Concepts:

- Conflict-free sets
- Defence of an argument
- Admissible sets

Extension-based semantics:

- Complete
- Grounded
- Preferred
- Stable
- Skeptical vs credulous acceptance

Introduction

Problems in the computation of semantics of argumentation frameworks can be divided into **enumeration** and **decision** problems.

Enumeration problems aim to compute one or all extensions of an argumentation framework under a particular semantics.

Decision problems have to do with determining whether a particular argument is credulously or skeptically accepted under a particular semantics.

Most computational approaches to these problems can be classified as either **reduction-based** or **direct**.

Introduction

Reduction-based approaches:

- Translate the argumentation problem into a different *equivalent* reasoning problem;
- Solve the translated problem;
- Translate the solution back into argumentation.

Examples include translating to: constraint satisfaction problem; answer set programming; and equational systems.

Big advantage in that we can exploit existing, well-developed and highly sophisticated solvers for the target translation problem domains.

Introduction

Direct approaches on the other hand solve the argumentation problem directly.

Advantage is that we can tailor the approaches specifically for the argumentation domain, incorporating short cuts specific to argumentation. We also don’t have the overhead of transformation.

Direct approaches include: labelling-based algorithms; dialectical argument games; decomposition variants.

Introduction

Direct approaches on the other hand solve the argumentation problem directly.

Advantage is that we can tailor the approaches specifically for the argumentation domain, incorporating short cuts specific to argumentation. We also don't have the overhead of transformation.

Direct approaches include: labelling-based algorithms; dialectical **argument games**; **decomposition** variants.

Introduction

If you're interested in finding out more about the different computational approaches to problem solving in argumentation, the following survey paper is an excellent place to start.

Günther Charwat, Wolfgang Dvořák, Sarah A. Gaggl, Johannes P. Wallner, Stefan Woltran. Methods for solving reasoning problems in abstract argumentation – A survey. In *Artificial Intelligence*, Volume 220, 2015, Pages 28-63, ISSN 0004-3702, https://doi.org/10.1016/j.artint.2014.11.008.

You could also look at the International Competition on Computational Models of Argumentation, where solvers compete on different argumentation framework benchmarks. http://argumentationcompetition.org

Argument game approach

Procedure to determine whether a given argument is part of an extension. We'll look at an approach for the grounded semantics.

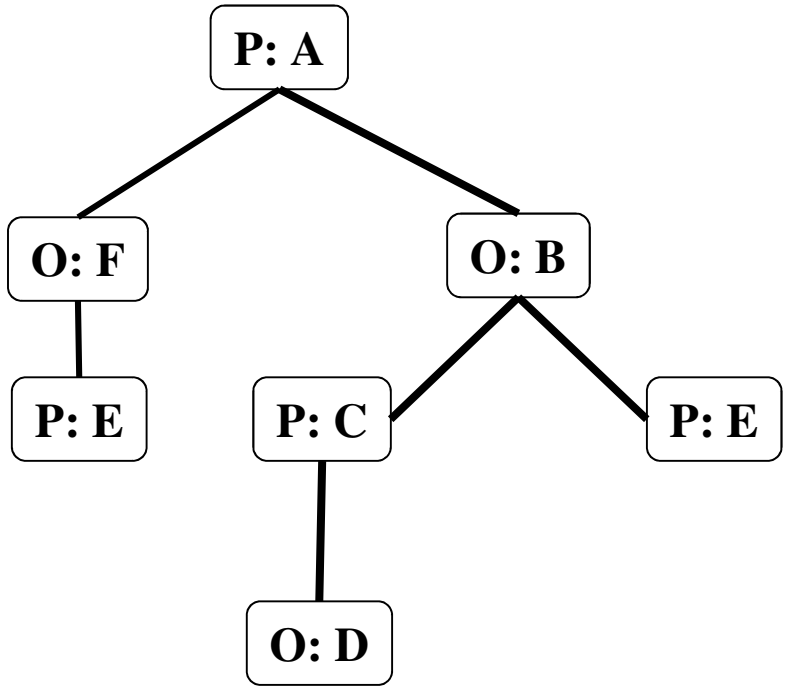
General idea:

- **Argument game** between **proponent** P and **opponent** O.
- Proponent starts with argument in question.
- Each party replies with suitable attacker (following *rules* of the game).
- We have some winning criterion (e.g., other player can't move).
- If P **can always** win, then the argument is in the extension:
  - means that there is a way P can play such that no matter what O does P will win; P has a **winning strategy**.

Argument game approach

A **game tree** represents all the possible games that can be played according to the rules of the game:

- Every branch is a **dispute** (sequence of allowable moves).
- The game tree contains all possible disputes.

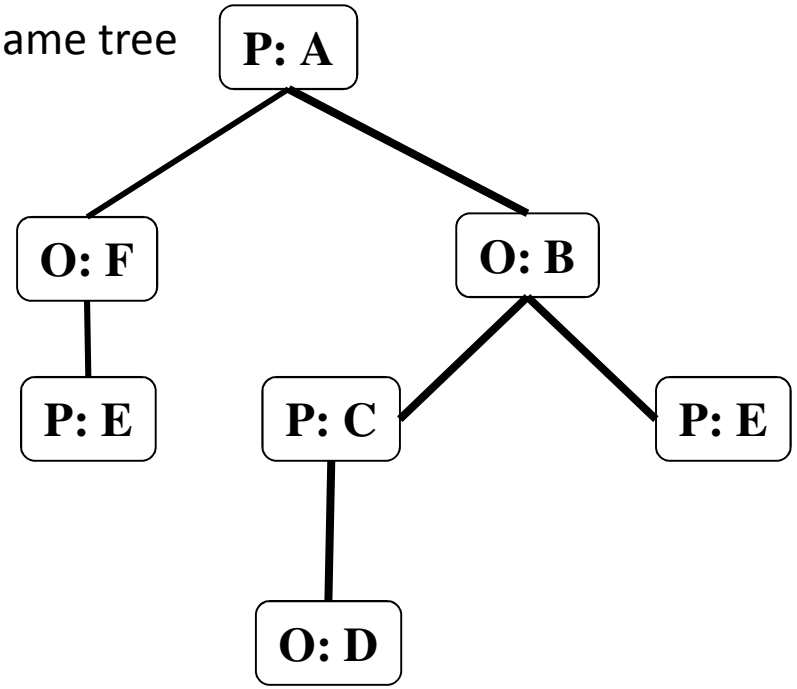


Argument game approach

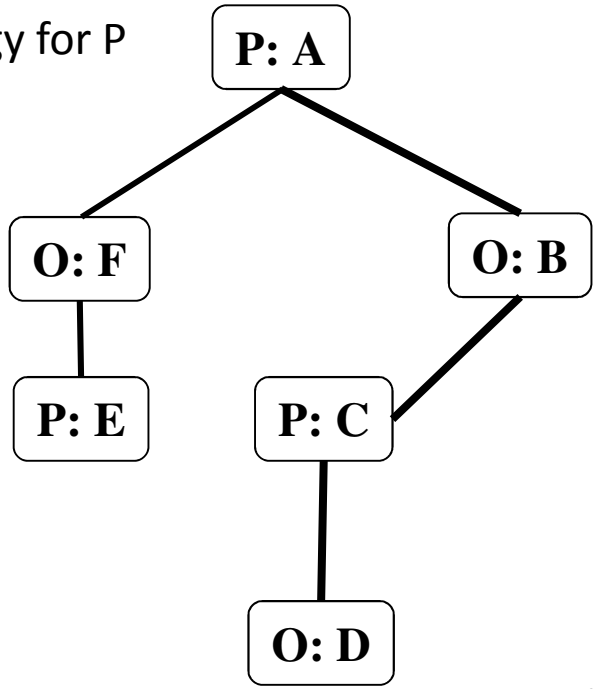
A **strategy** for player *p* is a partial game tree:

- Every branch is a **dispute** (sequence of allowable moves).
- The tree only branches after moves by *p*.
- The children of *p*'s moves are all the legal moves by the other player.

Game tree



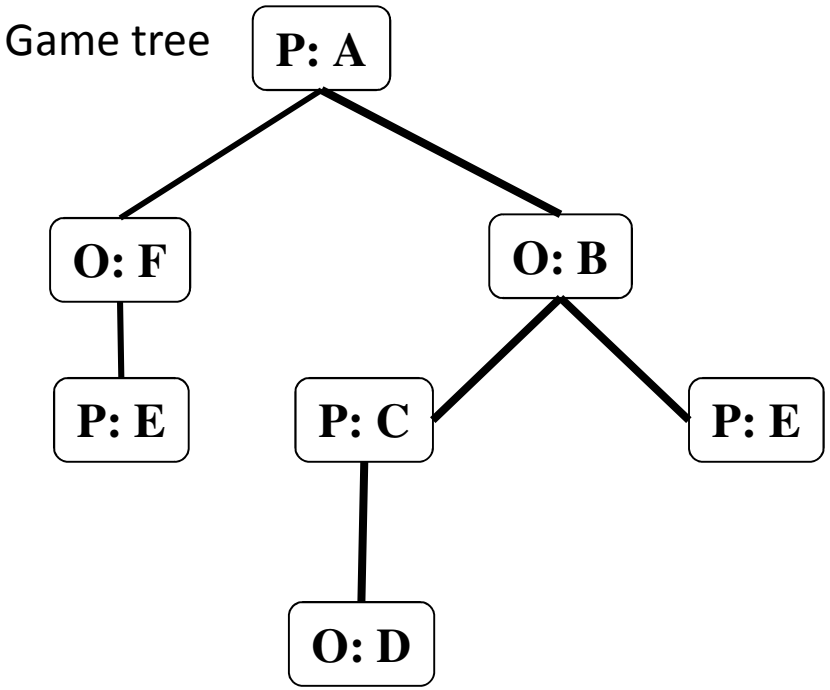
Strategy for P

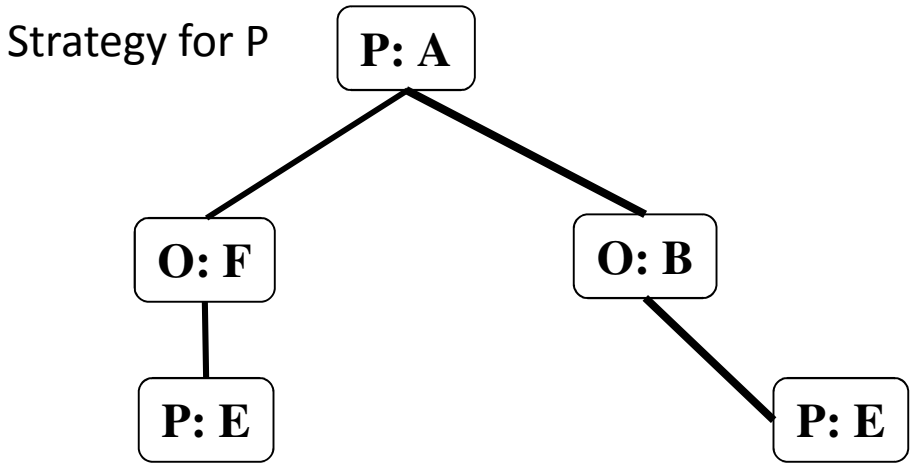


Argument game approach

A **strategy** for player *p* is a partial game tree:

- Every branch is a **dispute** (sequence of allowable moves).
- The tree only branches after moves by *p*.
- The children of *p*'s moves are all the legal moves by the other player.



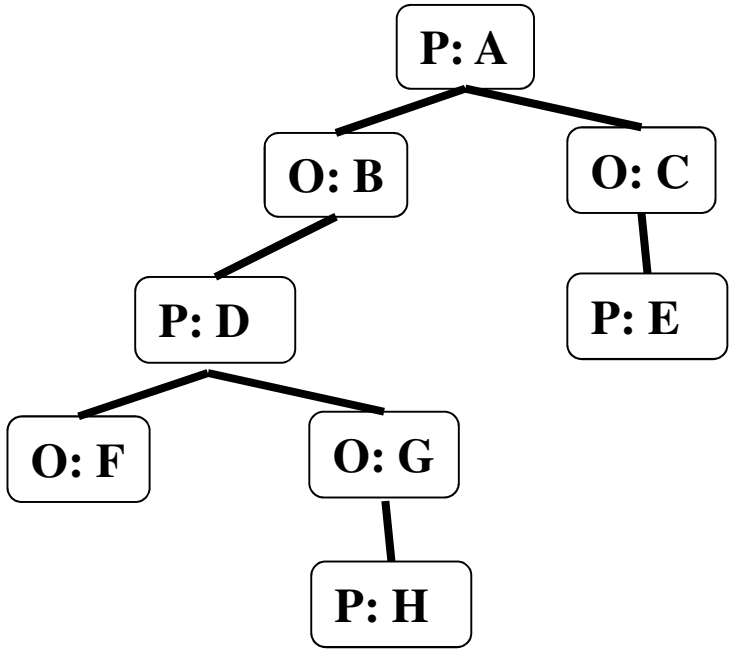


Argument game approach

A **strategy** for player *p* is a partial game tree:

- Every branch is a **dispute** (sequence of allowable moves).
- The tree only branches after moves by *p*.
- The children of *p*'s moves are all the legal moves by the other player.

This is a strategy for P.



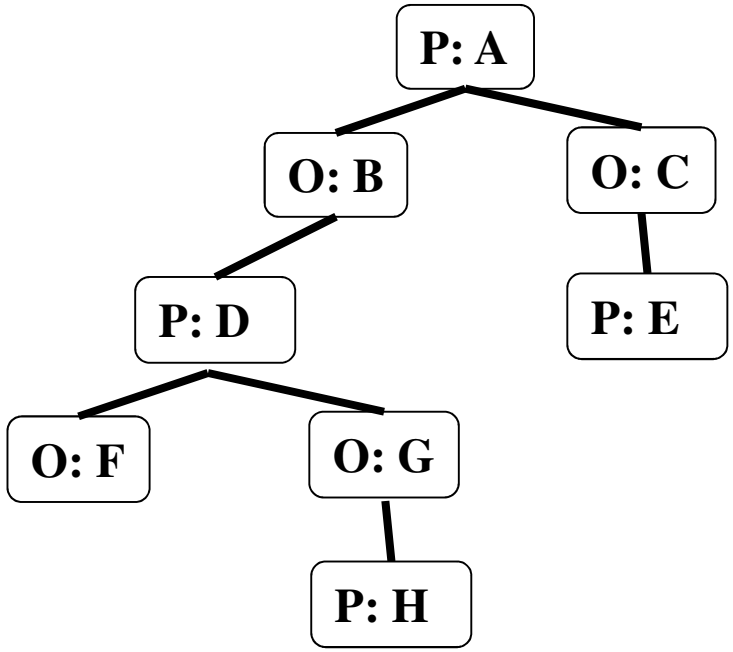
Argument game approach

A **strategy** for player *p* is a partial game tree:

- Every branch is a **dispute** (sequence of allowable moves).
- The tree only branches after moves by *p*.
- The children of *p*'s moves are all the legal moves by the other player.

This is a strategy for P.

Is it a strategy for O?



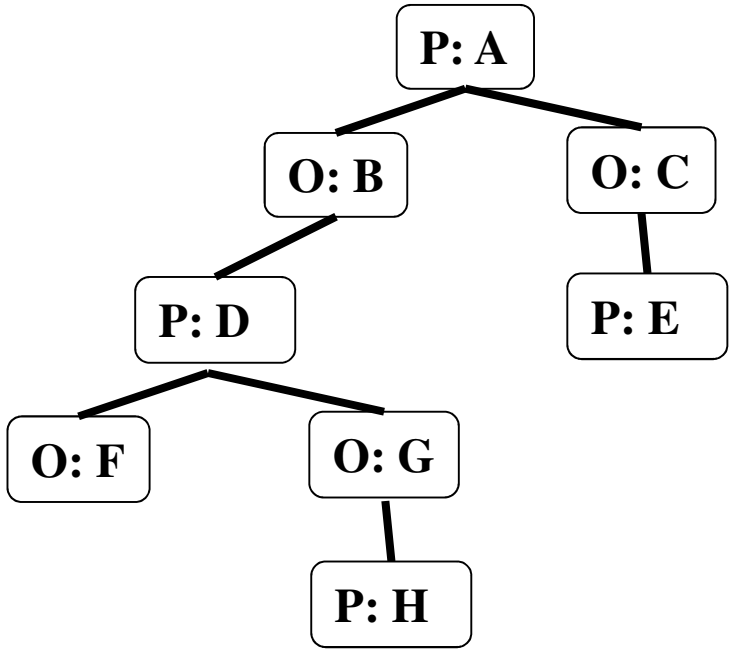
Argument game approach

A **strategy** for player *p* is a partial game tree:

- Every branch is a **dispute** (sequence of allowable moves).
- The tree only branches after moves by *p*.
- The children of *p*'s moves are all the legal moves by the other player.

This is a strategy for P.

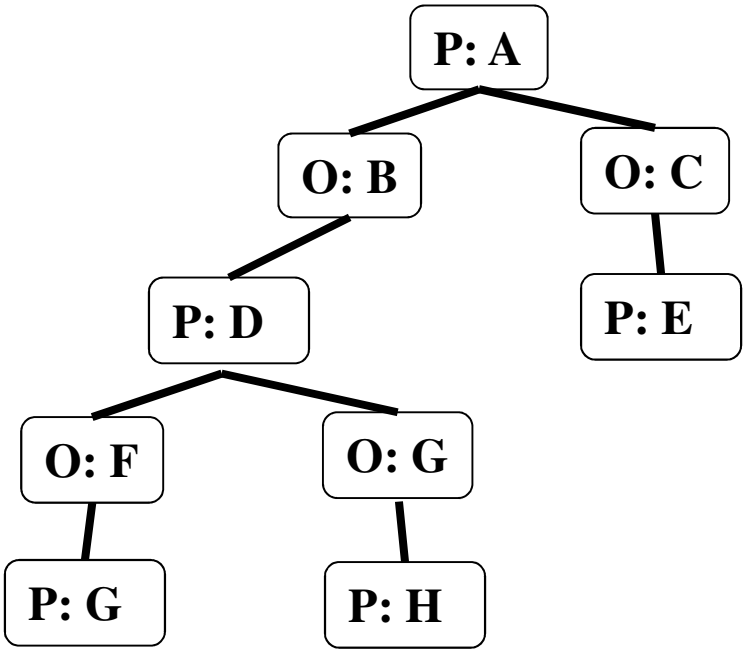
It is not a strategy for O, because it does not define the moves that O should make (it branches after P's moves).



Argument game approach

A **strategy** *S* for player *p* is **winning** if and only if *p* wins all disputes in *S.*

Idea: We want to define the rules of the game so that P has a winning strategy for a game where it begins with argument A if and only if A is in the grounded extension.



Argument game approach

The appropriate rules of the game and winning criterion depend on the semantics.

- May players *repeat their own* arguments?
- May players *repeat each other's* arguments?
- May they use *weakly attacking* arguments (i.e., where the attack is bi-directional)?
- May they *backtrack* (i.e., do they have to attack the previous argument moved or can they target an argument from earlier in the game)?

Argument game for grounded semantics

Each move must reply to the previous move (ie, backtracking not allowed).

Proponent cannot repeat moves.

Proponent moves strict attackers (can't be bi-directional attack).

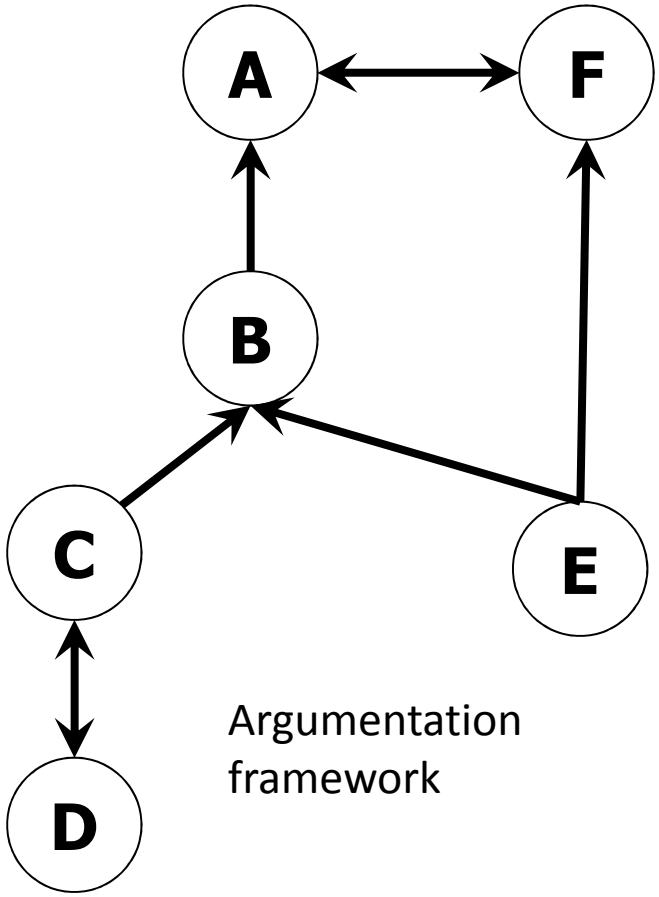
Opponent moves attackers (bi-directional attack ok).

A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

That is to say, the argument game for grounded semantics is sound and complete w.r.t. the decision problem of whether the argument is in the grounded extension.

Argument game for grounded semantics



Argumentation framework

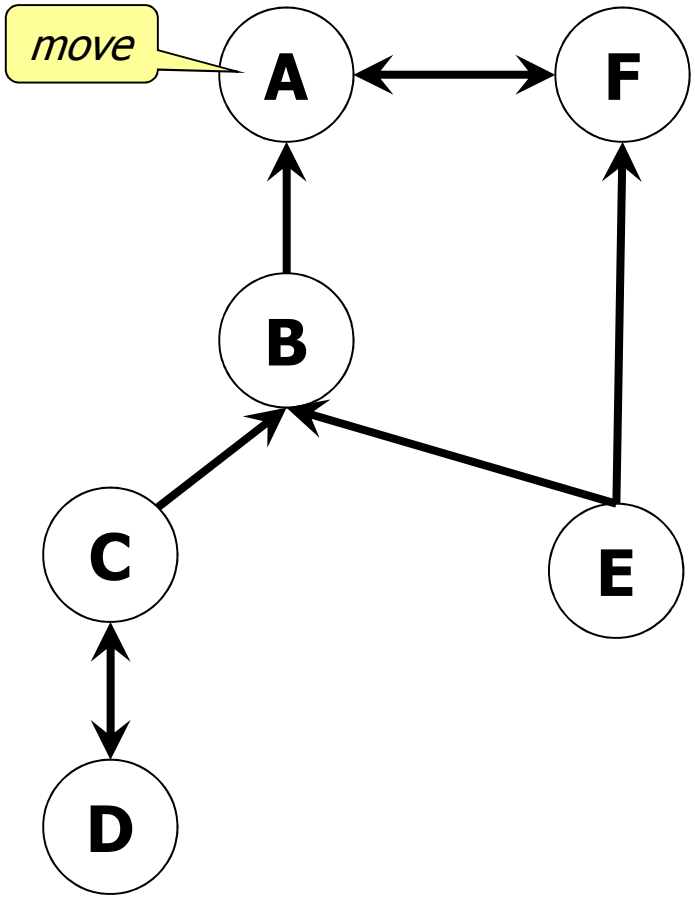
Example

Let's first build the game tree, that captures all the possible games we might have according to the rules of the game.

Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

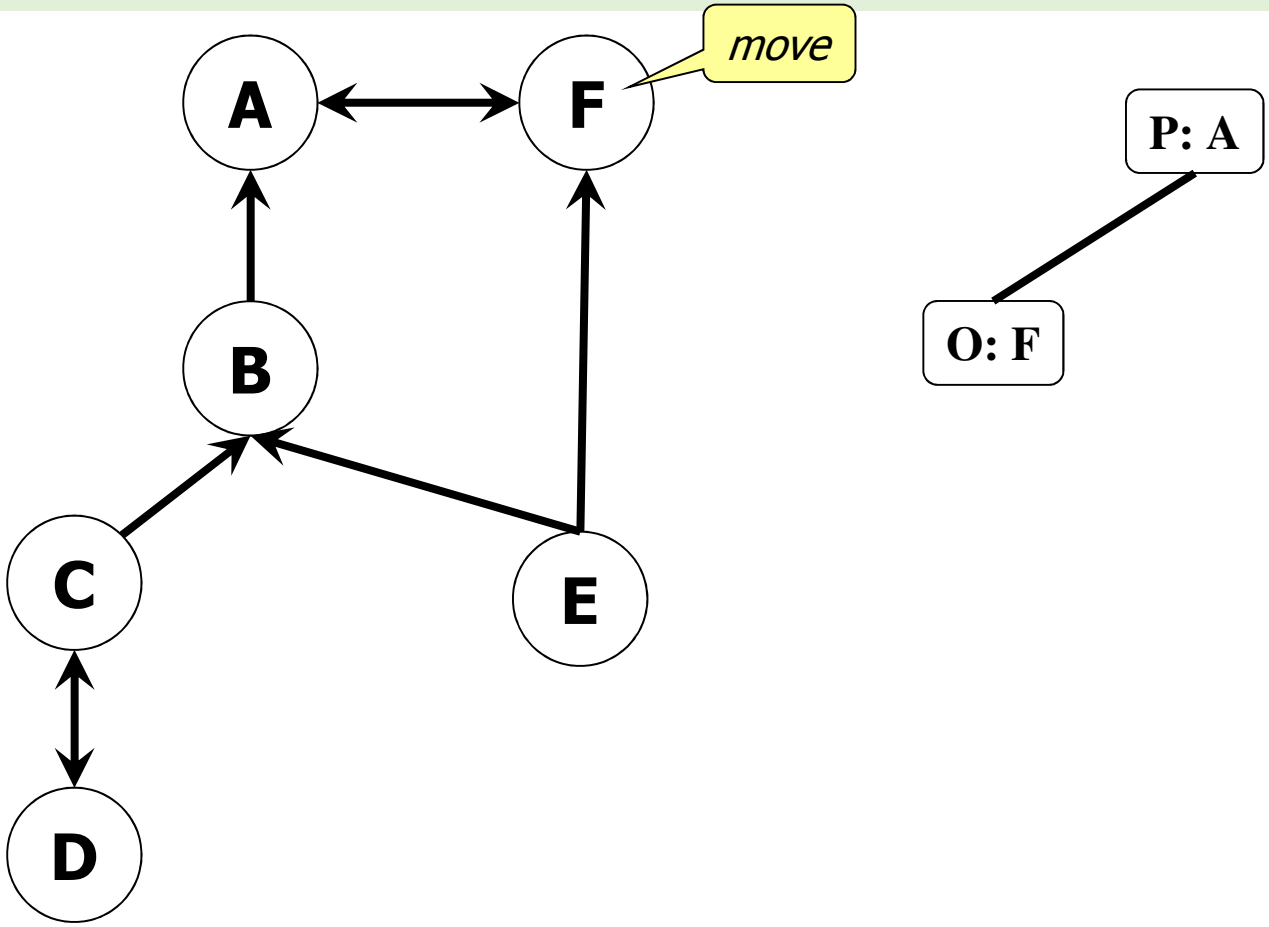


P: A

Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

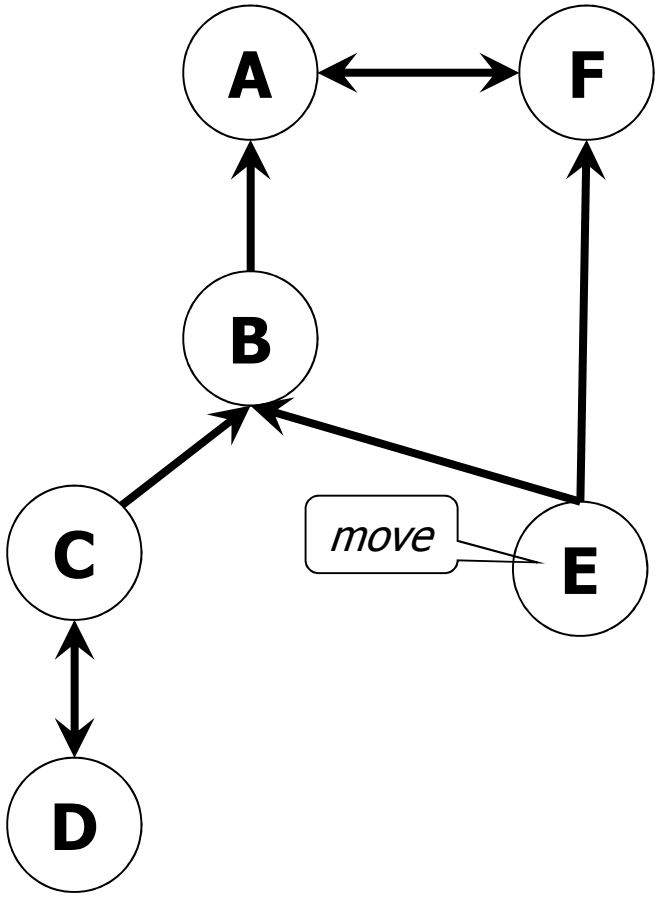
Argument game for grounded semantics

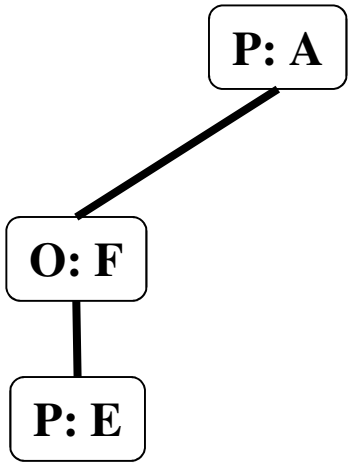


Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

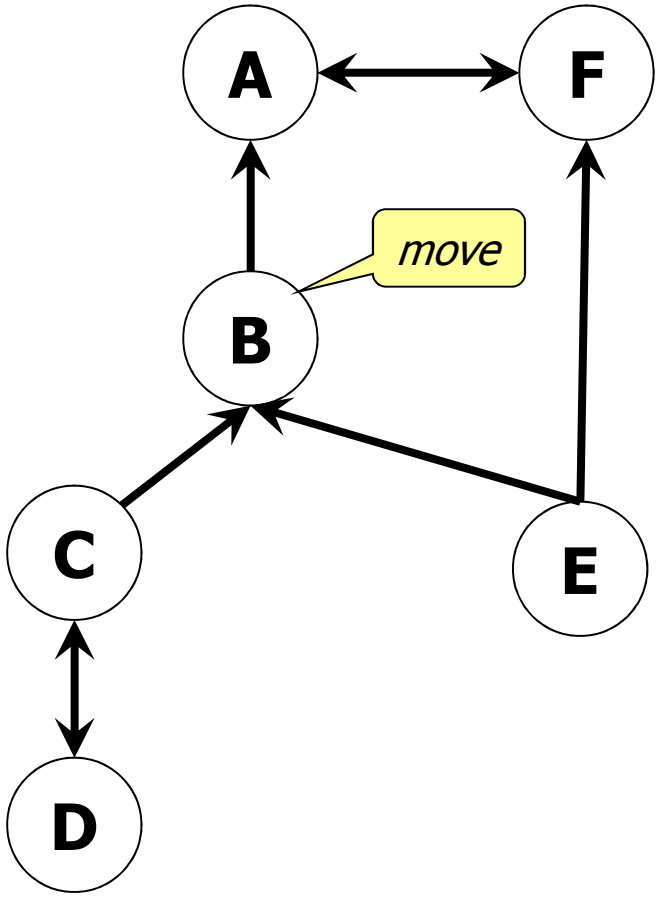


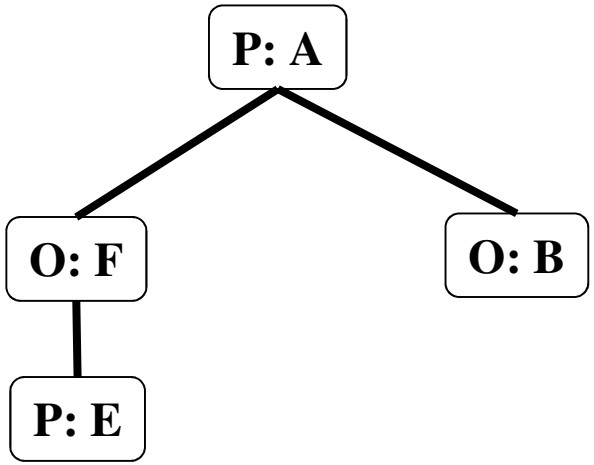


Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

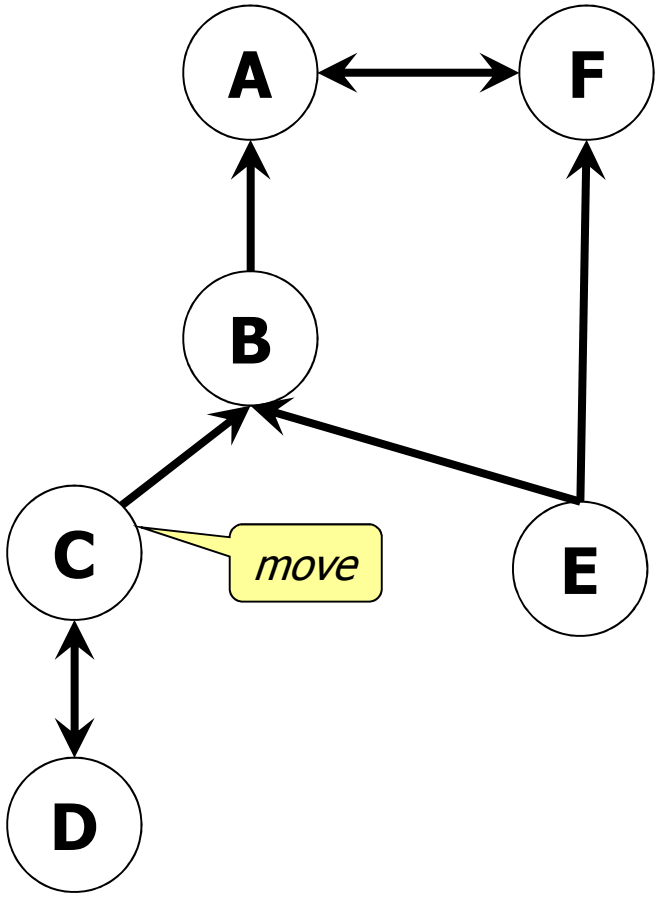


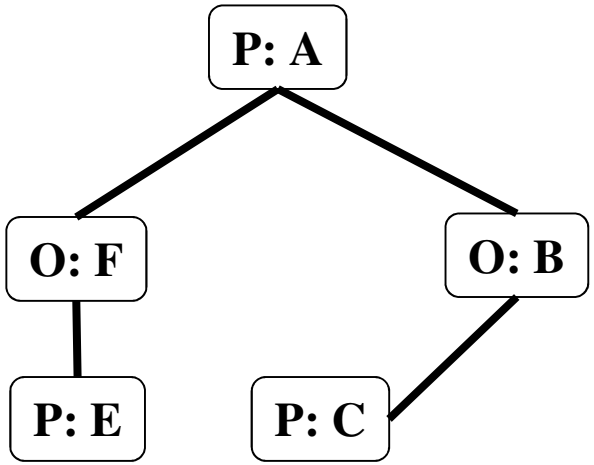


Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

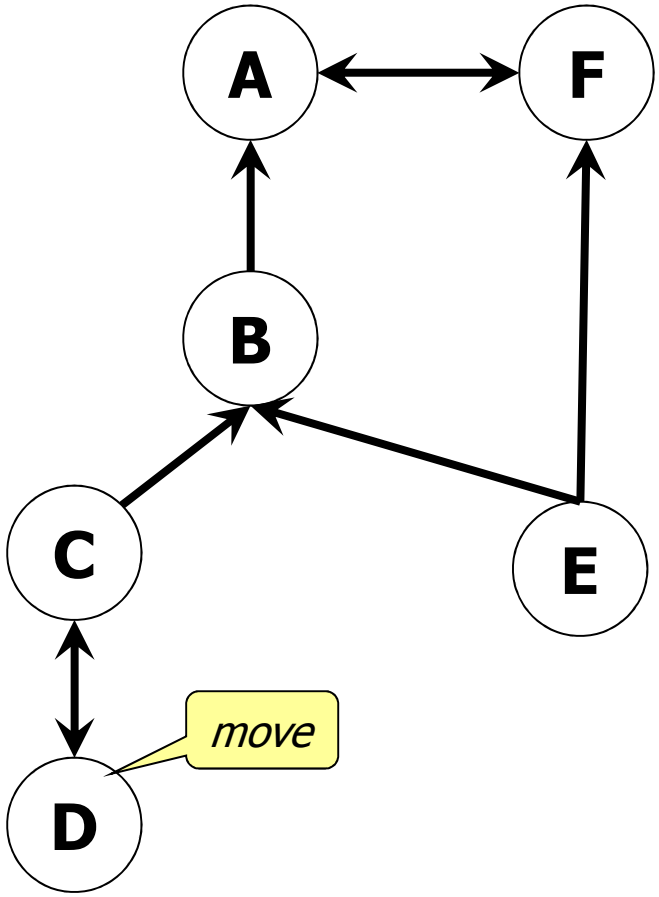


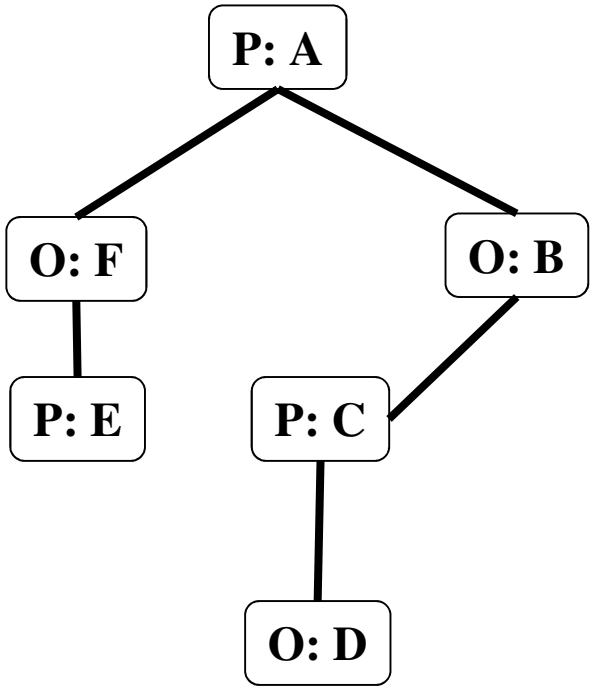


Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

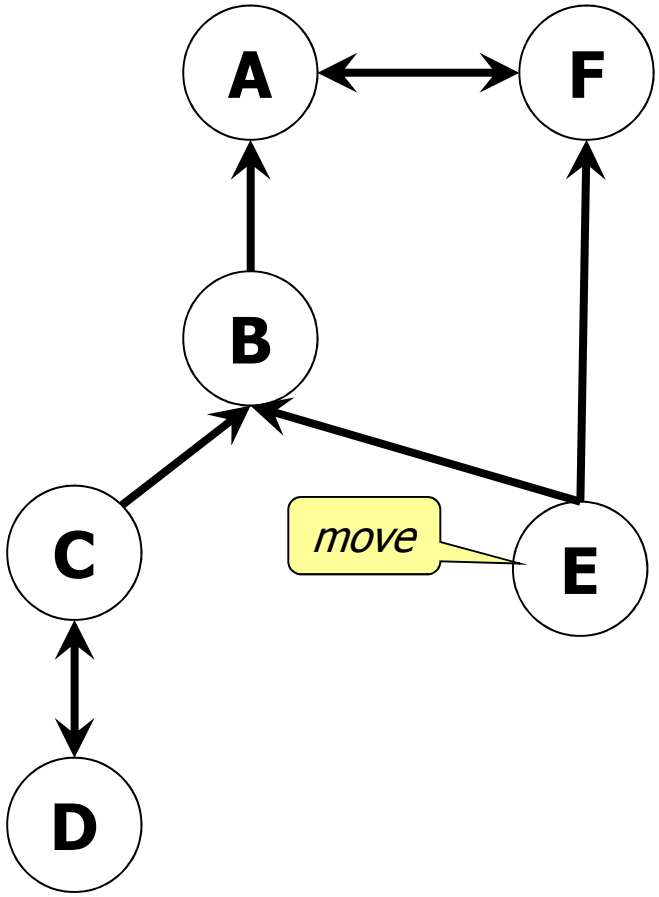


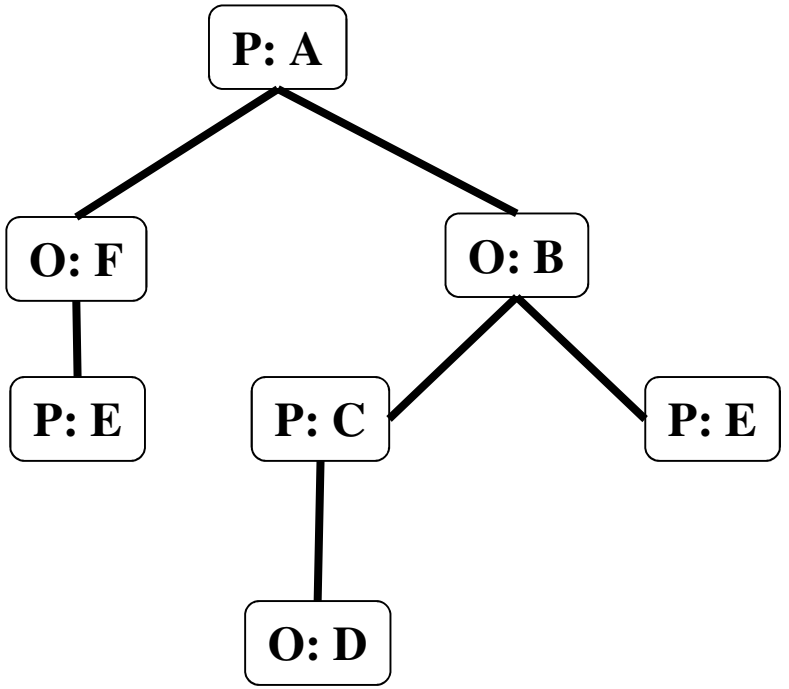


Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

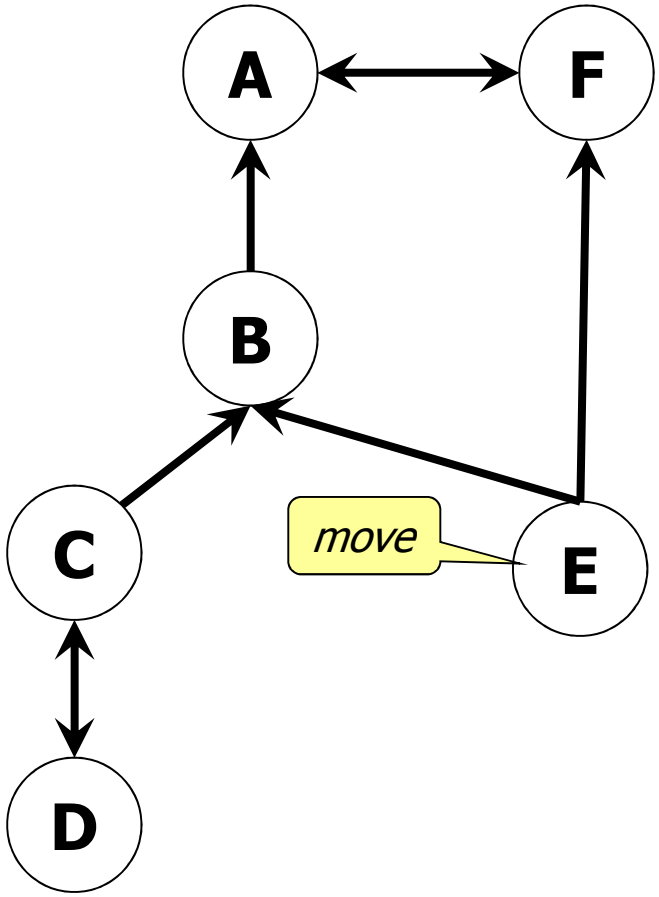


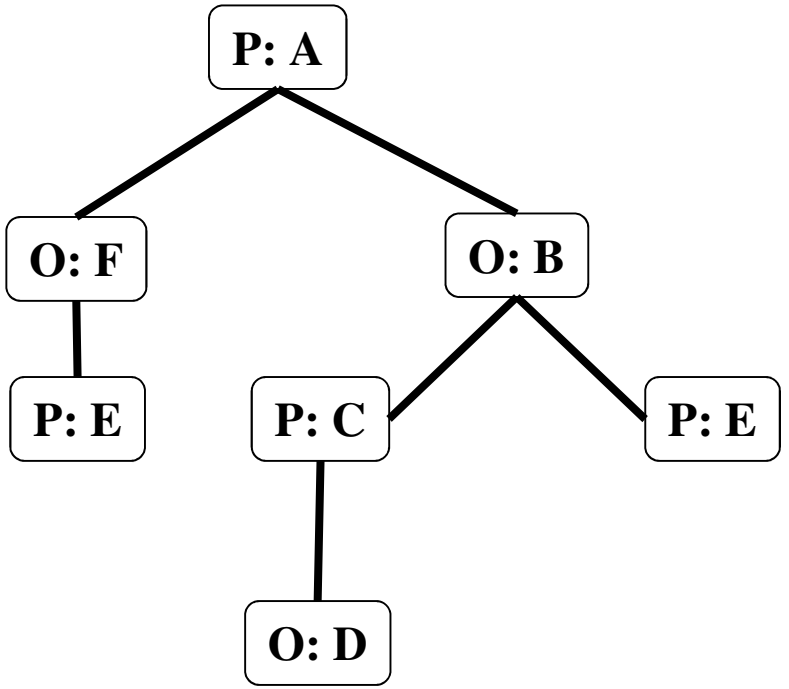


Each move must reply to the previous move. Proponent cannot repeat moves. Proponent moves strict attackers. Opponent moves attackers. A player wins if and only if the other player cannot move.

Then argument A is in the grounded extension if and only if the proponent has a winning strategy for the game that starts with the proponent moving the argument A.

Argument game for grounded semantics

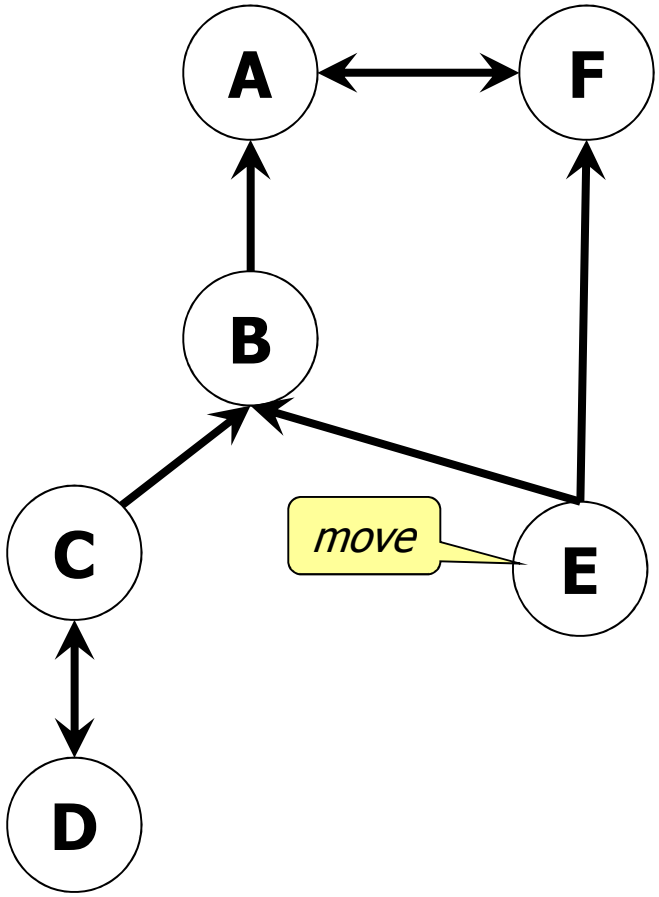


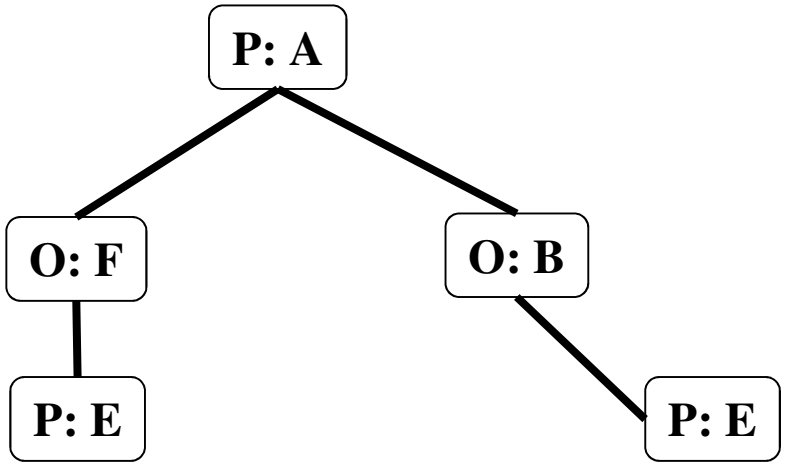


Game tree (on the right) captures all the allowed disputes.

Is there a *winning strategy for P*?

Argument game for grounded semantics



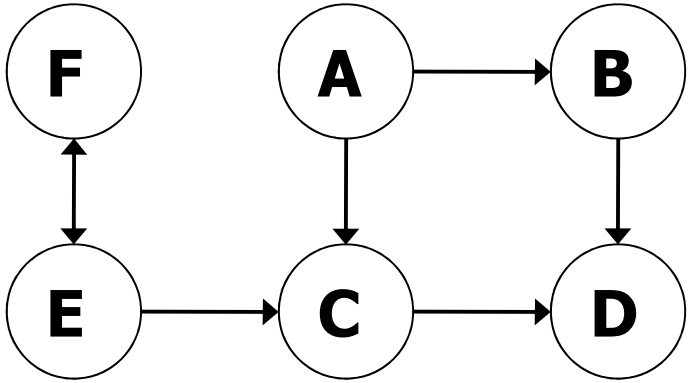


This (on right) is a **winning strategy** for P.

Argument game for grounded semantics

Exercise

- Draw the complete game tree for the argument D.
- How many strategies are there for P?
- How many strategies are there for O?
- Who has a winning strategy?

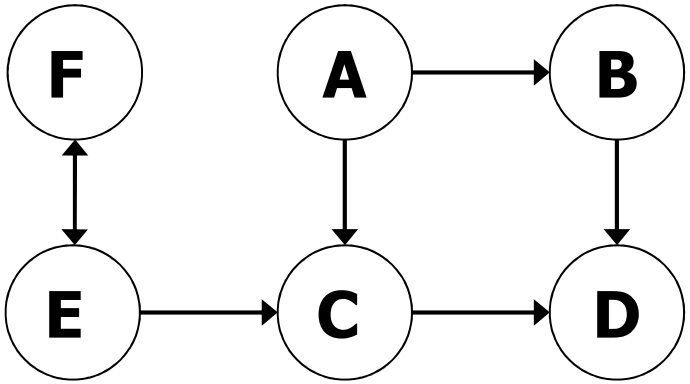


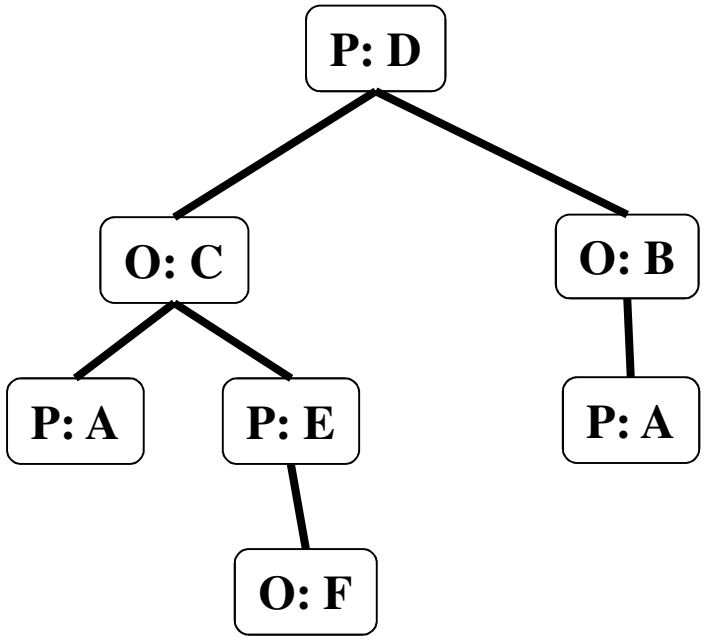
Argumentation framework

Remember:

- Proponent starts with argument D.
- Each move must reply to the previous move.
- Proponent cannot repeat moves.
- Proponent moves strict attackers, opponent moves attackers.
- A player wins if and only if the other player cannot move.

Argument game for grounded semantics

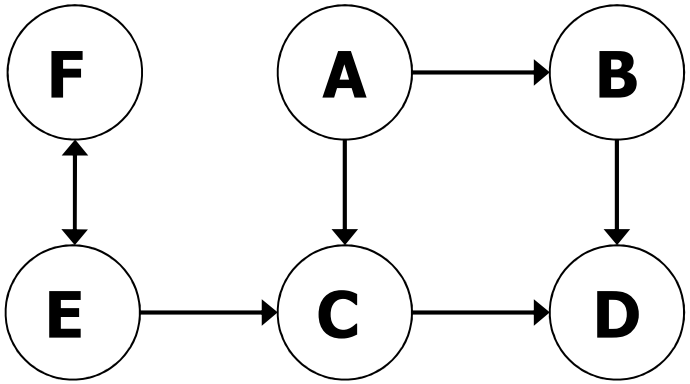


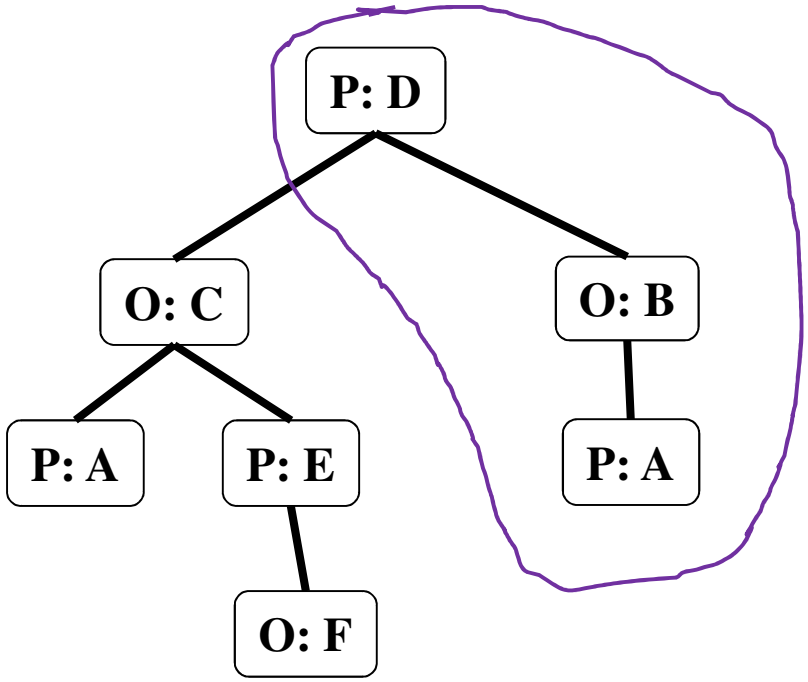


Remember:

- Proponent starts with argument D.
- Each move must reply to the previous move.
- Proponent cannot repeat moves.
- Proponent moves strict attackers, opponent moves attackers.
- A player wins if and only if the other player cannot move.

Argument game for grounded semantics





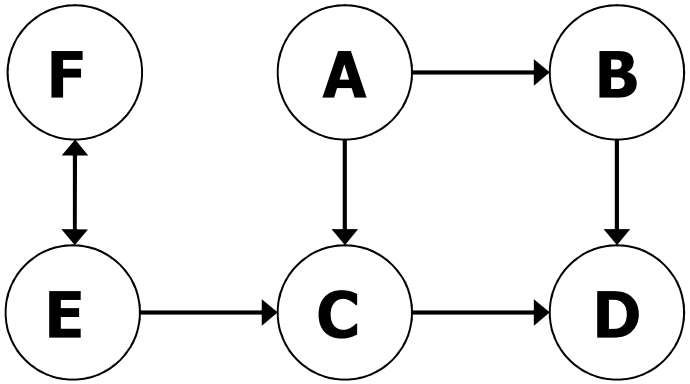
Remember:

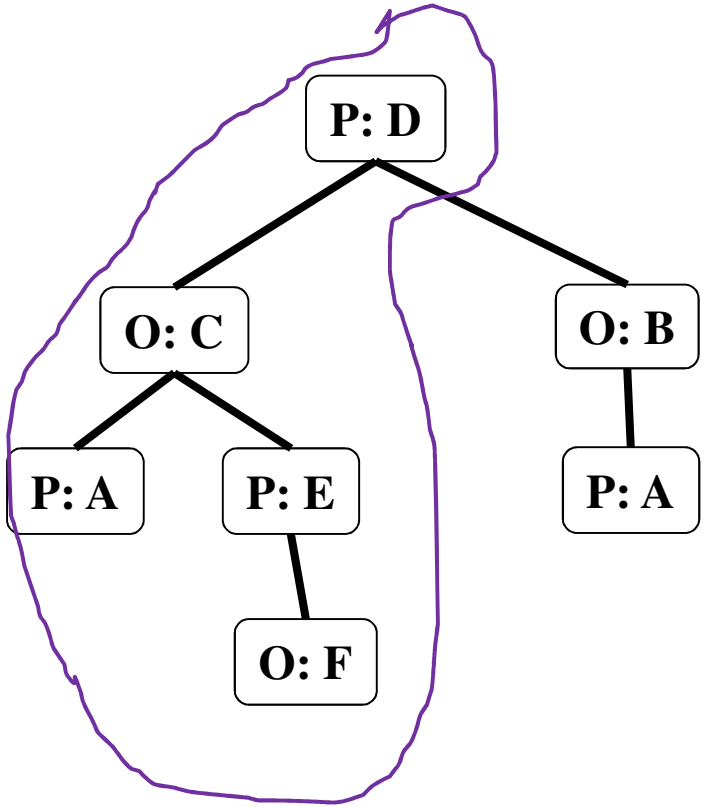
- Proponent starts with argument D.
- Each move must reply to the previous move.
- Proponent cannot repeat moves.
- Proponent moves strict attackers, opponent moves attackers.
- A player wins if and only if the other player cannot move.

Strategy for O.

Not a winning strategy.

Argument game for grounded semantics



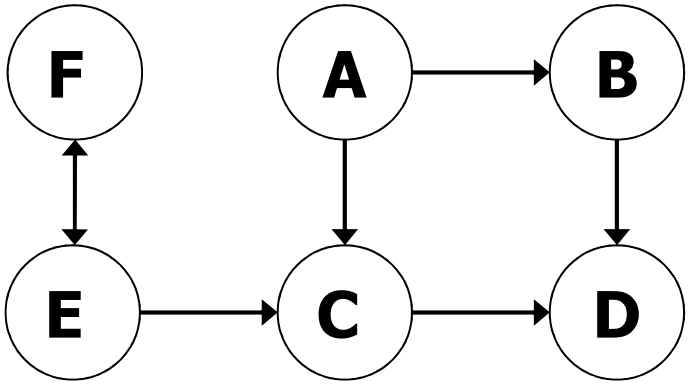


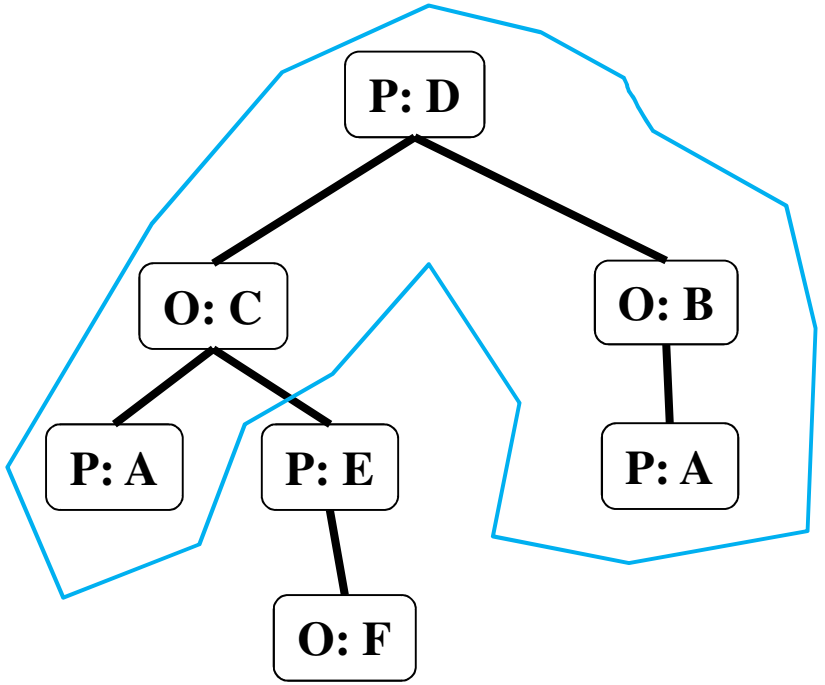
Strategy for O.
Not a winning strategy.

Remember:

- Proponent starts with argument D.
- Each move must reply to the previous move.
- Proponent cannot repeat moves.
- Proponent moves strict attackers, opponent moves attackers.
- A player wins if and only if the other player cannot move.

Argument game for grounded semantics



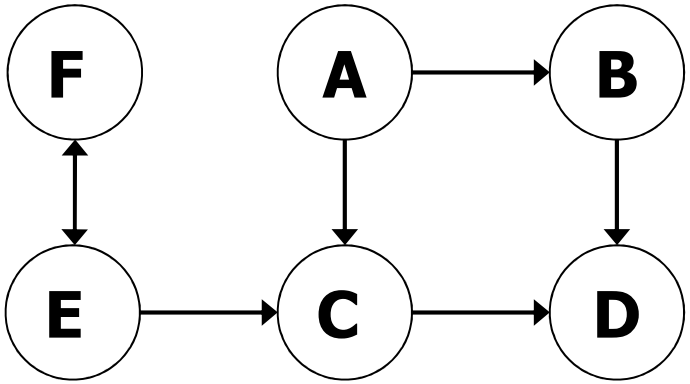


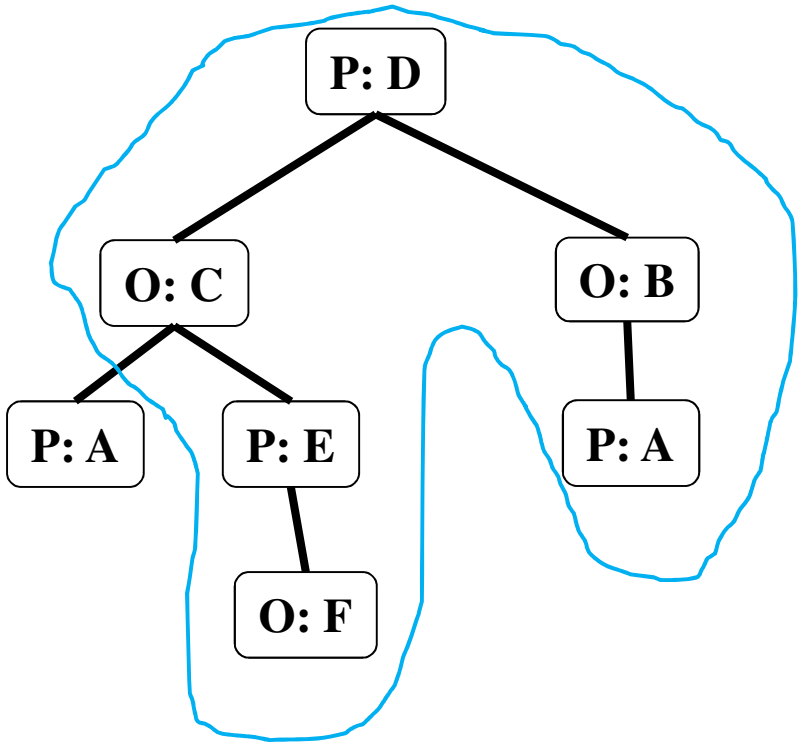
Remember:

- Proponent starts with argument D.
- Each move must reply to the previous move.
- Proponent cannot repeat moves.
- Proponent moves strict attackers, opponent moves attackers.
- A player wins if and only if the other player cannot move.

Winning strategy for P.
Means D is in the grounded extension.

Argument game for grounded semantics





Strategy for P.
Not a winning strategy.

Remember:

- Proponent starts with argument D.
- Each move must reply to the previous move.
- Proponent cannot repeat moves.
- Proponent moves strict attackers, opponent moves attackers.
- A player wins if and only if the other player cannot move.

Game-Theoretic Semantics

Converting an argumentation graph into a dialogue game between two players is an example of a Game-Theoretic Semantics:

To every statement X we associate a formal game G(X) between two players, Proponent P and Opponent O.

A statement X is true

if and only if

The game G(X) associated to X between two players P and O has a winning strategy for player P.

Examples:

- Ehrenfeucht-Fraïssé Games (aka Back-and-Forth Games)
- Independence-Friendly (IF) Logic.

Summary

This lecture we looked at a method that can be used to compute semantics:

- Argument game approach for determining membership of grounded extension

Another approach is presented in the Appendix.

- Decomposition approach for determining preferred extensions.

Next week, we give a brief introduction to Machine Learning.