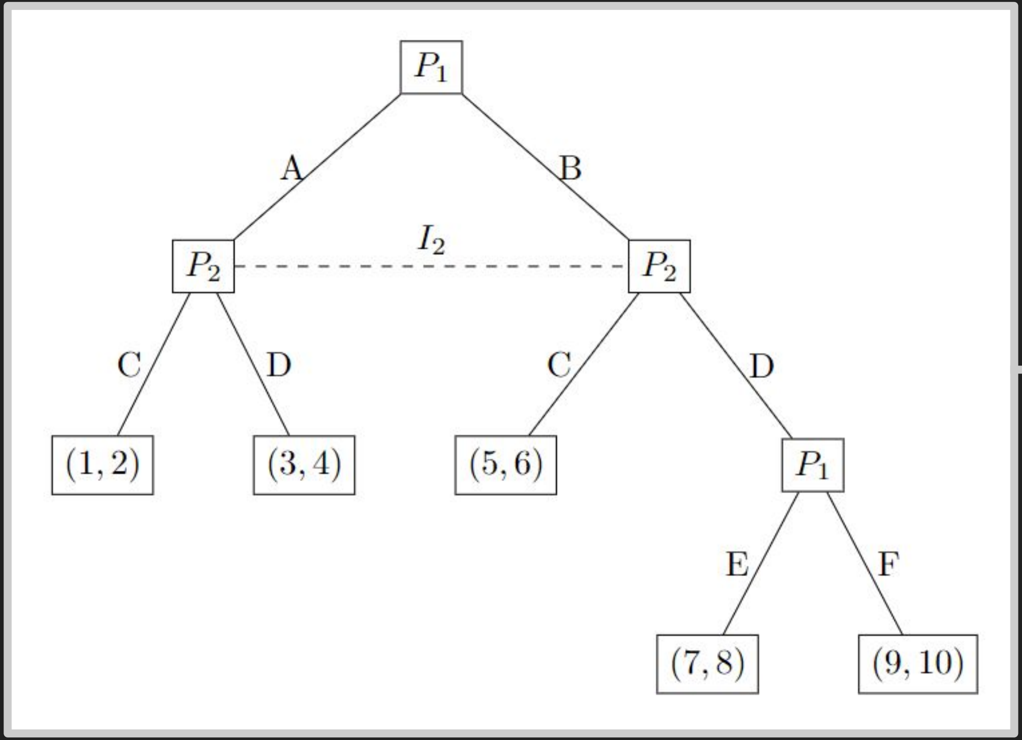
Sequence Form

Presenter: Brandon Dos Remedios

Major Points

1) The problem sequence form solves
2) The 4 components of sequence form
3) What property does it introduce that helps computationally?
4) What are realization plans and how can we best implement them?
5) How can put this into something computable

Consider a Small Game

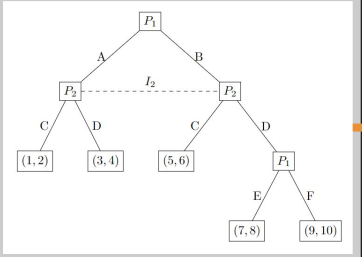


Properties:

- Extensive Form
- Imperfect Information
- Perfect Recall

The Problem

Initial Approach: Induced Normal Form (INF)



Extensive Form

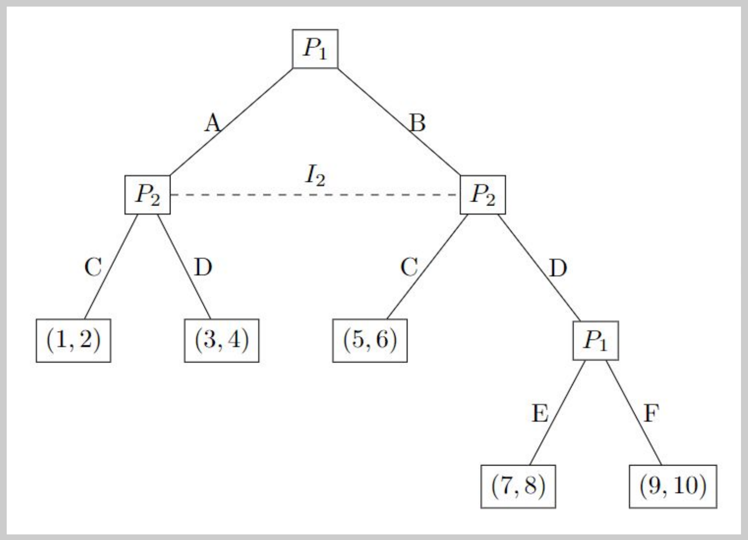
1) AE, BE, AF, BF
2) C, D

Pure Strategies

	C	D
AE	(1, 2)	(3, 4)
BE	(5, 6)	(7, 8)
AF	(1, 2)	(3, 4)
BF	(5, 6)	(9, 10)

INF Payoff Matrix

The Problem with INF



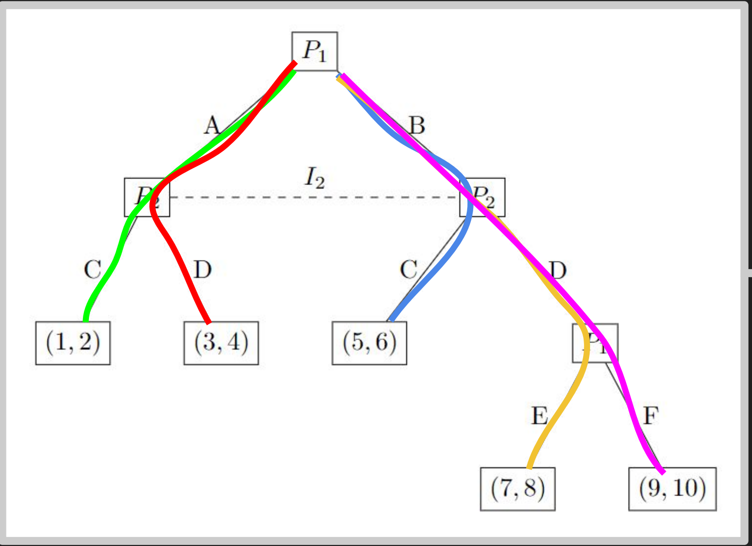
Number of pure strategies is the size of the cross product of actions at each node:

$$\implies O(e^n)$$

Sequence Form

An Initial Intuition

Instead of pure strategies consider the paths to leaf nodes in the tree



Number of paths to leaves is the number of leaves:

$$\implies O(n)$$

4 Components of Sequence Form

$$G = (N, \Sigma, g, C)$$

Agents

Set of Sequences

$$\Sigma = \Sigma_1 \times \dots \times \Sigma_n$$

Payoff Function

$$g = (g_1, \dots, g_n)$$

Constraints

$$C = (C_1, \dots, C_n)$$

Formal Definition: Sequences and Payoff

Sequence

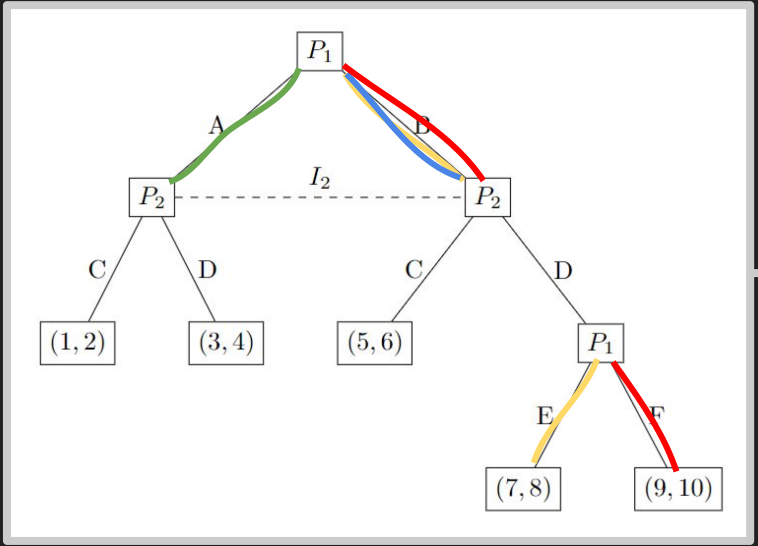
- Defined for a player *i* for some node $h \in H \cup Z$
- A **sequence** is an ordered set of player *i*'s actions that lie on the path to *h*
- **Set of sequences for player i Σ_i**, is the set of player *i* sequences that lead to a node for player *i*
- Sequence to root is Ø

Payoff

- Defined for a player *i* for some member σ of the set of all sequences Σ

$$g_i(\sigma) = \begin{cases} u_i(z) & \text{if } \sigma \text{ legally reaches leaf node } z \\ 0 & \text{otherwise} \end{cases}$$

Our Small Game’s Sequences



1: {Ø, A, B, BE, BF}
2: {Ø, C, D}

Our Small Game's INF vs Sequence Form: Payoff Matrices

	C	D
AE	(1, 2)	(3, 4)
BE	(5, 6)	(7, 8)
AF	(1, 2)	(3, 4)
BF	(5, 6)	(9, 10)

INF

	∅	C	D
∅	(0, 0)	(0, 0)	(0, 0)
A	(0, 0)	(1, 2)	(3, 4)
B	(0, 0)	(5, 6)	(0, 0)
BE	(0, 0)	(0, 0)	(7, 8)
BF	(0, 0)	(0, 0)	(9, 10)

Sequence Form

INF vs Sequence Form: Payoff Matrices

	C	D
AE	(1, 2)	(3, 4)
BE	(5, 6)	(7, 8)
AF	(1, 2)	(3, 4)
BF	(5, 6)	(9, 10)

8 Non-zeroes
Not Sparse

	∅	C	D
∅			
A		(1, 2)	(3, 4)
B		(5, 6)	
BE			(7, 8)
BF			(9, 10)

5 Non-zeroes
Sparse!

Sparsity Advantage

- Large research base in taking advantage of sparseness for computation
- **Sparse** when non-zeros are **O(n+m)** instead of **O(nm)** which would be **dense**
- Key Idea: Ignore computations when things will obviously result in 0, reduces the amount we have to do

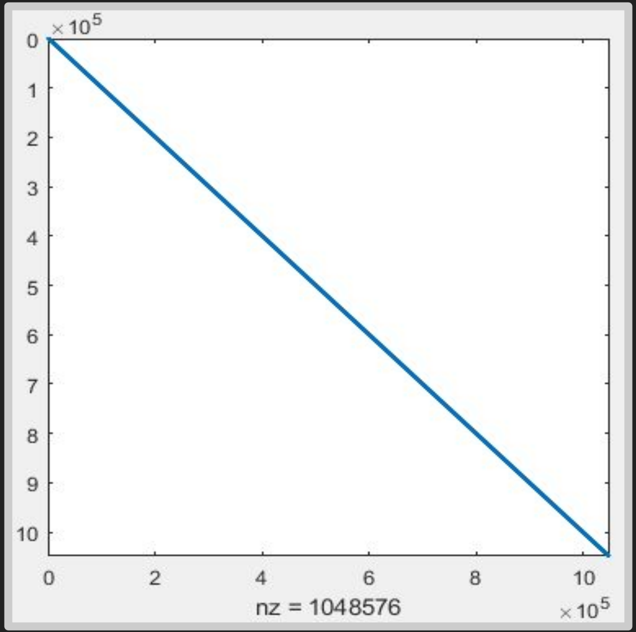
	C	D
AE	(1, 2)	(3, 4)
BE	(5, 6)	(7, 8)
AF	(1, 2)	(3, 4)
BF	(5, 6)	(9, 10)

	∅	C	D
∅			
A		(1, 2)	(3, 4)
B		(5, 6)	
BE			(7, 8)
BF			(9, 10)

A Really Quick Example

Sparse

```
>> spy(speye(2^20))
```



Not Sparse

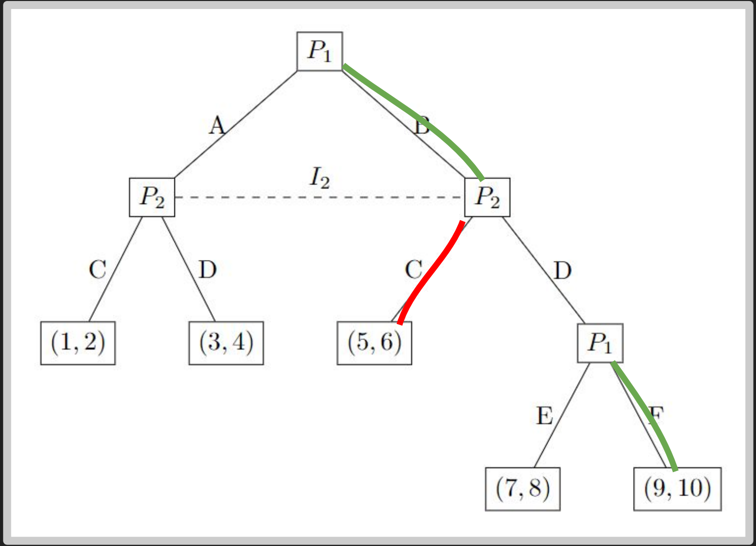
```
>> spy(eye(2^20))
```

```
Error using eye
Requested 1048576x1048576 (8192.0GB) array excee
cause MATLAB to become unresponsive.
```

Realization Plans

Sequences Aren’t Enough

- Sequences can’t take the place of actions entirely
- Still need to assign what to do at every node (want a behavioral strategy)



Behavioral Strategy to Realization Plan

- Behavioral strategy since perfect recall guarantees an equilibrium (Kuhn, 1953 + Nash, 1951)
- Assignment of some probability to every choice node h for player *i*, of taking some action at that node:

$$\beta_i(h, a_i)$$

- Realization plan:
  - Defined for a behavioral strategy β_i

$$r_i : \Sigma_i \to [0, 1] \qquad r_i(\sigma_i) = \prod_{(h,a_i)\in\sigma_i} \beta_i(h, a_i)$$

Linear Constraint Definition

$$\mathrm{seq}_i : I_i \rightarrow \Sigma_i$$

- Maps information set to the sequence that leads to it

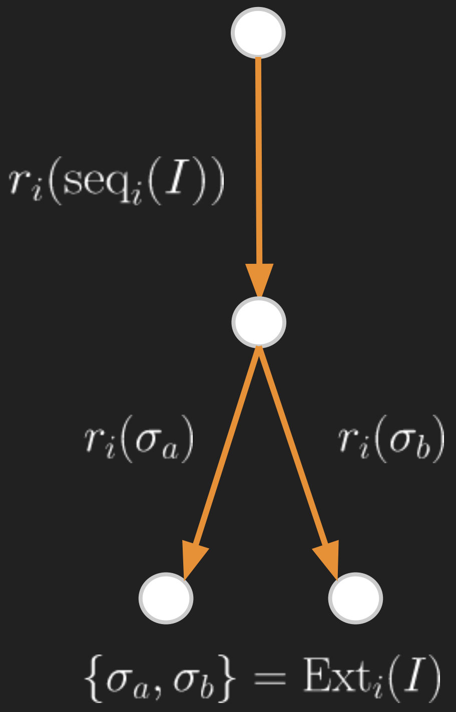
$$\mathrm{Ext}_i : \Sigma_i \rightarrow 2^{\Sigma_i}$$

- Maps sequence to sequences that extend it

Define realization plan within linear constraints using these:

$$r_i(\varnothing) = 1,\ r_i(\sigma_i) \geq 0,\ \forall \sigma_i \in \Sigma_i$$

$$\sum_{\sigma_i' \in \mathrm{Ext}_i(I)} r_i(\sigma_i') = r_i(\mathrm{seq}_i(I)),\ \forall I \in I_i$$



Realization Plan Back to Behavioral Strategy

$$\beta_i(h, a_i) \equiv \frac{r_i(\mathrm{seq}_i(I)a_i)}{r_i(\mathrm{seq}_i(I))}$$

- This is equivalent to what we did earlier:

$$\prod_{(h,a_i)\in\sigma_i} \beta_i(h, a_i) = \frac{r_i(\mathrm{seq}_i(I)a_i)}{r_i(\mathrm{seq}_i(I))} \cdot \frac{r_i(\mathrm{seq}_i(I))}{r_i(\mathrm{seq}_i(I)a_{-i})} \cdot \ldots \cdot \frac{r_i(\mathrm{seq}_i(I)a_{-n})}{r_i(\varnothing)} = r_i(\mathrm{seq}_i(I)a_i)$$

Linear Programming and Duality

Primal Linear Programming

- Constraints are linear and we can define an objective
- Variable: Realization plan
- Consider a 2-person game, the Primal LP of agent 1*’s* best response given agent 2’s realization plan is as follows:

$$\max \sum_{\sigma_1 \in \Sigma_1, \sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2) r_1(\sigma_2)$$

$$\text{subj. to } \; r_1(\varnothing) = 1, \; r_1(\sigma) \geq 0 \; \forall \sigma \in \Sigma_1$$

$$\sum_{\sigma \in \mathrm{Ext}_1(I)} r_1(\sigma) = r_1(\mathrm{seq}_1(I)) \; \forall I \in I_1$$

Primal Linear Programming

- Constraints are linear and we can define an objective
- Variable: Realization plan
- Consider a 2-person game, the Primal LP of agent 1's best response given agent 2's realization plan is as follows:

$$\max \sum_{\sigma_1 \in \Sigma_1, \sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) \boxed{r_2(\sigma_2) r_1(\sigma_2)}$$

$$\text{subj. to } \; r_1(\varnothing) = 1, \; r_1(\sigma) \geq 0 \; \forall \sigma \in \Sigma_1$$

$$\sum_{\sigma \in \mathrm{Ext}_1(I)} r_1(\sigma) = r_1(\mathrm{seq}_1(I)) \; \forall I \in I_1$$

Quadratic if both are variables

LP Duality

- Can get an equivalent problem to a Primal LP problem

$$\begin{aligned} \max \ & c^T x \\ \text{subj. to} \ & Ax = b \\ & x \geq 0 \end{aligned} \quad \Longleftrightarrow \quad \begin{aligned} \min \ & y^T b \\ \text{subj. to} \ & A^T y - z = c \\ & z \geq 0 \end{aligned}$$

Primal LP **Dual LP**

LP Dual Process

Step 1: Loosen Restrictions into Objective

$$c^T x \rightarrow c^T x + y^T (b - Ax)$$

Step 2: Optimize the bound

$$\begin{cases} y^T b & \text{if } c - A^T y \leq 0 \\ \infty & \text{if } c - A^T y > 0 \end{cases}$$

Step 3: Convert into equivalent Dual LP

$$\begin{aligned} \min \quad & y^T b \\ \text{subj. to} \quad & A^T y - z = c \\ & z \geq 0 \end{aligned}$$

An Equivalent Dual LP (Computable in Zero-sum)

- Linear w.r.t the variables:

$$\begin{aligned} \min \;\; & v_o \\ \text{subj. to} \;\; & v_{I_1(\sigma_1)} - \sum_{I' \in I_1(\mathrm{Ext}_1(\sigma_1))} v_{I'} \geq \sum_{\sigma_2 \in \Sigma_2} g_1(\sigma_1, \sigma_2) r_2(\sigma_2) \;\; \forall \sigma_1 \in \Sigma_1 \end{aligned}$$

- For zero-sum/constant-sum can insert constraints on player 2’s realization plan and optimize w.r.t those and the constraints on player 1’s primal

$$r_2(\varnothing) = 1, r_2(\sigma_2) \geq 0 \; \forall \sigma_2 \in \Sigma_2$$
$$\sum_{\sigma_2' \in \mathrm{Ext}_2(I)} r_2(\sigma_2') = r_2(\mathrm{seq}_2(I)) \; \forall I \in I_2$$

Computational Advantage/Limitations

- With this formulation we get a linear number of variables and constraints as well as a sparsity within the constraints
- Simplex method for solving LPs is potentially exponential w.r.t the variables and constraints, so we can’t say anything super strong, but this is as least as good as before

Major Takeaways

- Using INF leads to a solution space with a exponential number of dimensions w.r.t the size of the extensive form game
- Sequence form reduces the size of the solution space, by considering sequences instead of pure strategies
- This introduces sparsity within payoffs reducing computation work
- Realization plans allow sequences to be converted into usable behavioral strategies and can be specified by linear constraints
- LP can be used to best compute equilibria with this framework for certain games, specifically using tools such as the Dual LP to get a linear objective

References

- (Shoham and Leyton-Brown, 2009, p.72, 134-142) *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*
- (von Stengel, 1994) *Efficient Computation of Behavior Strategies*
- (Ascher and Grief, p. 271-286) *A First Course in Numerical Methods: Ch 9.3: Constrained Optimization*
- (Ascher and Grief, Unpublished/Currently Being Written Sorry) *Unpublished Second Edition, Ch: Sparse Direct Solvers*