CPS 590.01
LP and IP in Game theory (Normal-form Games, Nash Equilibria and Stackelberg Games)

Joshua Letchford

Zero-sum game (Mini-max)

		Them	
		L	R
Us	L	1, -1	-1, 1
	R	-1, 1	1, -1

- Assume opponent knows our **mixed** strategy
- If we play L 50%, R 50%...
- … opponent will be indifferent between R and L…
- … we get .5*(-1) + .5*(1) = 0

General-sum games

- You could still play a minimax strategy in general-sum games
  - I.e., pretend that the opponent is only trying to hurt you
- But this is not rational:

0, 0	3, 1
1, 0	2, 1

- If Column was trying to hurt Row, Column would play Left, so Row should play Down
- In reality, Column will play Right (strictly dominant), so Row should play Up
- Is there a better generalization of minimax strategies in zero-sum games to general-sum games?

Nash equilibrium

[Nash 50]

- A vector of strategies (one for each player) is called a strategy profile
- A strategy profile $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ is a Nash equilibrium if each σ_i is a best response to σ_{-i}
  - That is, for any i, for any σ_i', $u_i(\sigma_i, \sigma_{-i}) \geq u_i(\sigma_i', \sigma_{-i})$
- Note that this does not say anything about multiple agents changing their strategies at the same time
- In any (finite) game, at least one Nash equilibrium (possibly using mixed strategies) exists [Nash 50]
- (Note - singular: equilibrium, plural: equilibria)

The presentation game

Presenter

Audience	*Put effort into presentation (E)*	*Do not put effort into presentation (NE)*
Pay attention (A)	4, 4	-16, -14
Do not pay attention (NA)	0, -2	0, 0

- Pure-strategy Nash equilibria: (A, E), (NA, NE)
- Mixed-strategy Nash equilibrium:

  ((1/10 A, 9/10 NA), (4/5 E, 1/5 NE))
  - Utility 0 for audience, -14/10 for presenter
  - Can see that some equilibria are strictly better for both players than other equilibria

Some properties of Nash equilibria

- If you can eliminate a strategy using strict dominance or even iterated strict dominance, it will not occur (i.e., it will be played with probability 0) in every Nash equilibrium
  - Weakly dominated strategies may still be played in some Nash equilibrium
- In 2-player zero-sum games, a profile is a Nash equilibrium if and only if both players play minimax strategies
  - Hence, in such games, if (σ_1, σ_2) and (σ_1', σ_2') are Nash equilibria, then so are (σ_1, σ_2') and (σ_1', σ_2)
    - No equilibrium selection problem here!

Solving for a Nash equilibrium using MIP (2 players)

[Sandholm, Gilpin, Conitzer AAAI05]

- maximize *whatever you like (e.g., social welfare)*
- subject to
  - for both i, $\Sigma_{s_i} \mathbf{p}_{s_i} = 1$
  - for both i, for all s_i, $\Sigma_{s_{-i}} \mathbf{p}_{s_{-i}} u_i(s_i, s_{-i}) = \mathbf{u}_{s_i}$
  - for both i, for all s_i, $\mathbf{u}_i \geq \mathbf{u}_{s_i}$ $\quad$ ($u_i = \max u_{s_i}$)
  - for both i, for all s_i, $\mathbf{p}_{s_i} \leq \mathbf{b}_{s_i}$
  - for both i, for all s_i, $\mathbf{u}_i - \mathbf{u}_{s_i} \leq M(1- \mathbf{b}_{s_i})$

- $\mathbf{b}_{s_i}$ is a binary variable indicating whether s_i is in the support, M is a large number

Stackelberg (commitment) games

	L	R
L	1, -1	3, 1
R	2, 1	4, -1

- Unique Nash equilibrium is (R,L)
  - This has a payoff of (2,1)

Commitment

	L	R
L	(1,-1)	(3,1)
R	(2,1)	(4,-1)

- What if the officer has the option to (credibly) announce where he will be patrolling?
- This would give him the power to “commit” to being at one of the buildings
  - This would be a pure-strategy Stackelberg game

Commitment…

	L	R
L	(1,-1)	(3,1)

- If the officer can commit to always being at the left building, then the vandal's best response is to go to the right building
  - This leads to an outcome of (3,1)

Committing to mixed strategies

	L	R
L	(1,-1)	(3,1)
R	(2,1)	(4,-1)

- What if we give the officer even more power: the ability to commit to a mixed strategy
  - This results in a mixed-strategy Stackelberg game
  - E.g., the officer commits to flip a weighted coin which decides where he patrols

Committing to mixed strategies is more powerful

	L	R
L	(1,-1)	(3,1)
R	(2,1)	(4,-1)

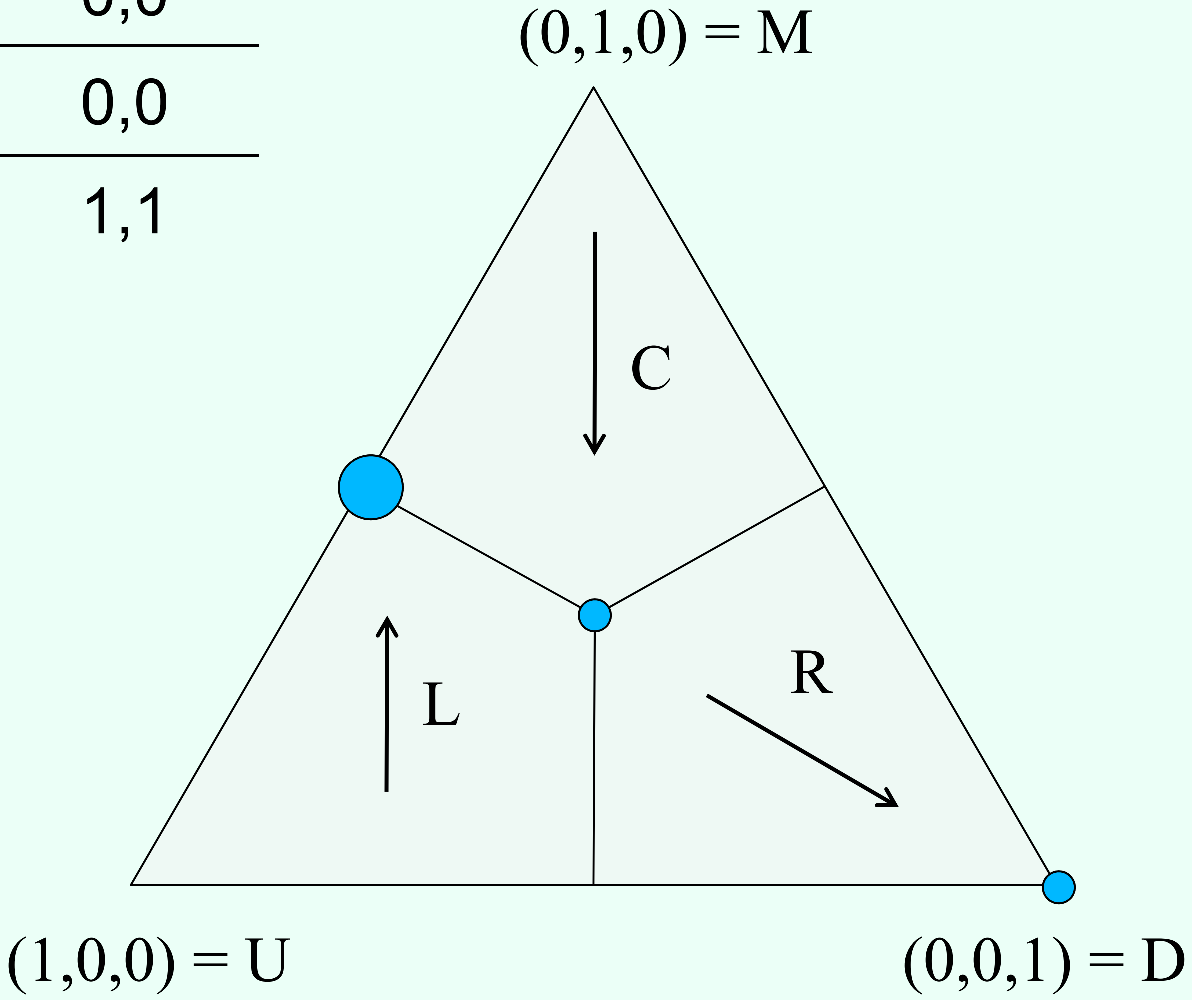
- Suppose the officer commits to the following strategy: {(.5+ε)L,(.5- ε)R}
  - The vandal's best response is R
  - As ε goes to 0, this converges to a payoff of (3.5,0)

Stackelberg games in general

- One of the agents (the leader) has some advantage that allows her to commit to a strategy (pure or mixed)
- The other agent (the follower) then chooses his best response to this

Visualization

	L	C	R
U	0,1	1,0	0,0
M	4,0	0,1	0,0
D	0,0	1,0	1,1



Easy polynomial-time algorithm for two players

[Conitzer & Sandholm EC'06, von Stengel & Zamir GEB'10]

- For every column j, we solve separately for the best mixed row strategy (defined by z_i) that induces player 2 to play j

$$\forall_j \; max \sum_i z_i R_{ij}$$

$$\forall_{j'} \sum_i z_i C_{ij} \geq \sum_i z_i C_{ij'}$$

$$\sum_i z_i = 1$$

I rows
J columns
R is the defenders payoff matrix
C is the attackers payoff matrix
z_i is the probability that row i is played

- (May be infeasible for some j)
- Pick the j that is best for player 1

Extensions

- A few extensions with LP or MIP formulations:
  - Bayesian setting (DOBSS [1])
    - Uses a MIP to avoid exponential size
  - Multiple Defense Resources (ERASER [2])
    - Assumes the structure is a “Security game”
    - Uses this structure to achieve a compact representation
  - Defense Costs [3]
    - Explicit costs for defense rather than limited defense resources

[1] Paruchuri et al. Playing Games for Security: An Efficient Exact Algorithm for Solving Bayesian Stackelberg Games
[2] Kiekintveld et al. Computing Optimal Randomized Resource Allocations for Massive Security Games
[3] Letchford and Vorobeychik. Computing Optimal Security Strategies for Interdependent Assets

(a particular kind of) Bayesian games

leader utilities

2	4
1	3

follower utilities (type 1)

1	0
0	1

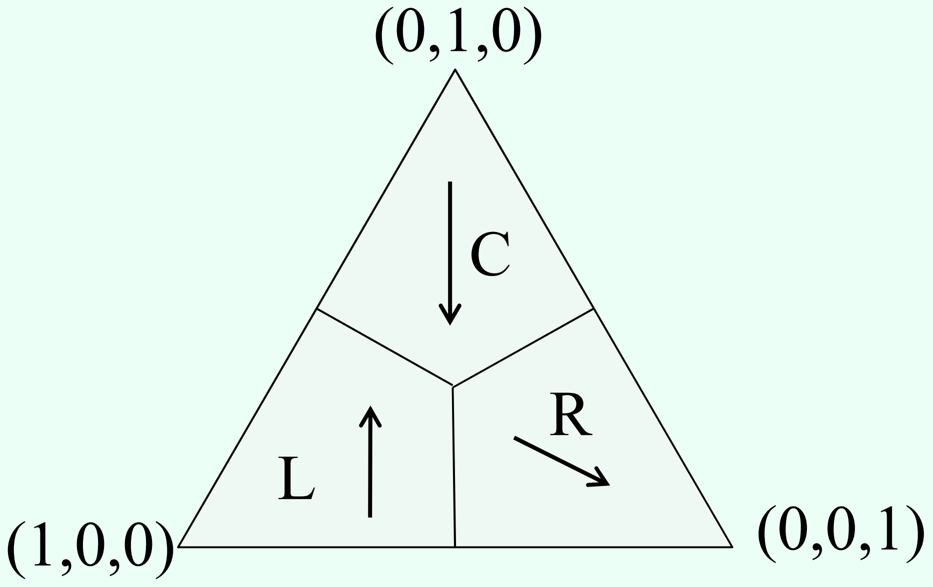
probability .6

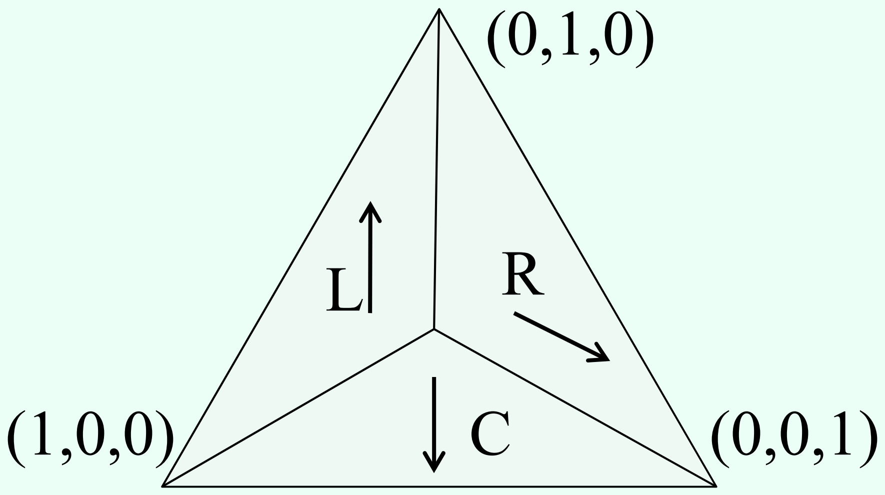
follower utilities (type 2)

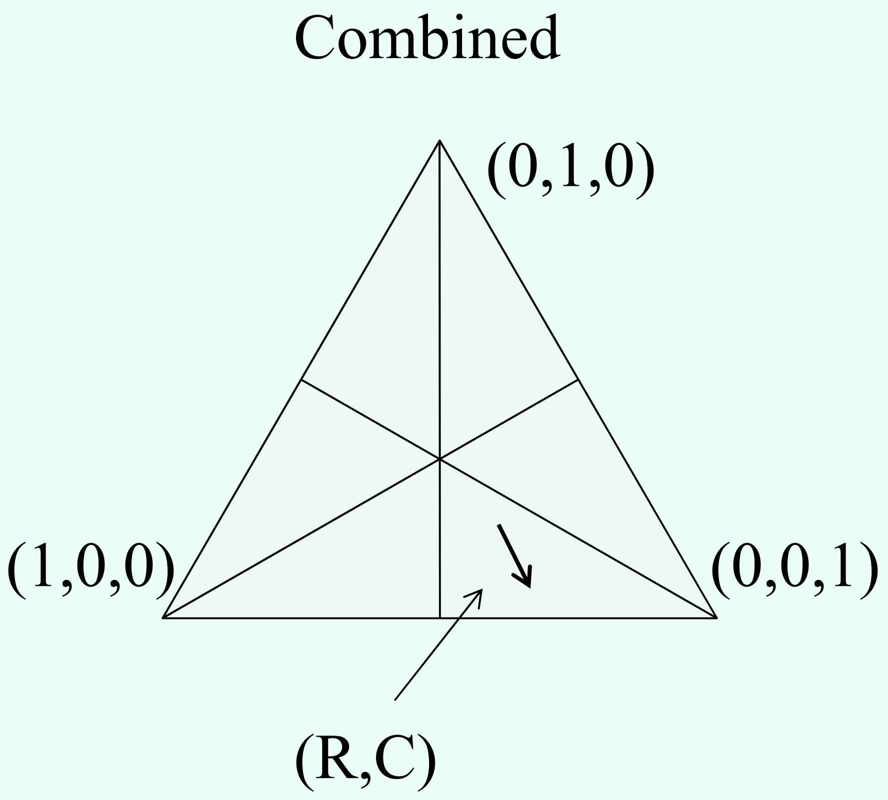
1	0
1	3

probability .4

Multiple types - visualization







DOBSS [Paruchuri et al. AAMAS'08] (MIP for the Bayesian setting)

$$max \sum_l \sum_i \sum_j p^l R^l_{ij} z^l_{ij}$$

$$\forall_l \sum_i \sum_j z^l_{ij} = 1$$

$$\forall_{l,j}\ q^l_j \leq \sum_i z^l_{ij} \leq 1$$

$$\forall_l \sum_j q^l_j = 1$$

$$\forall_{l,j}\ 0 \leq (a^l - \sum_i C^l_{ij}(\sum_{j'} z^l_{ij'})) \leq (1 - q^l_j)M$$

$$\forall_{l,i} \sum_j z^l_{ij} = \sum_j z^1_{ij}$$

I rows
J columns
p^l the probability that type *l* appears
R^l is the defenders payoff matrix
C^l is the attackers payoff matrix
z^l_{ij} is the probability that row *i* and column *j* are played against type *l*
q^l_j=1 when type *l*'s best response is column j
M is a large number

$$\forall_{l,i,j}\ z^l_{ij} \in [0, 1]$$

$$\forall_{l,j}\ q^l_j \in \{0, 1\}$$

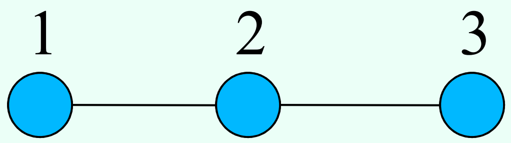
(In)approximability of Bayesian games [Letchford et al. SAGT'09]

- (# types)-approximation: optimize for each type separately using the LP method. Pick the solution that gives the best expected utility against the entire type distribution.

- Can't do any better in polynomial time, unless P=NP
  - Reduction from INDEPENDENT-SET

- For adversarially chosen types, cannot decide in polynomial time whether it is possible to guarantee positive utility, unless P=NP
  - Again, a MIP formulation can be given

Reduction from independent set



leader utilities

	A	B
a_l^1	1	0
a_l^2	1	0
a_l^3	1	0

follower utilities (type 1)

	A	B
a_l^1	3	1
a_l^2	0	10
a_l^3	0	1

follower utilities (type 2)

	A	B
a_l^1	0	10
a_l^2	3	1
a_l^3	0	10

follower utilities (type 3)

	A	B
a_l^1	0	1
a_l^2	0	10
a_l^3	3	1

Security games [Kiekintveld et al. AAMAS'09]

- Makes a simple assumption, namely that payoffs only depend on the identify of the target attacked and if that target is defended or not.
  - Often combined with an assumption that the defender is always better off when the attacked target is defended, and the attacker is always better off when the attacked target is undefended.

	Defended	Undefended
T_1	(5,-10)	(-20,30)
T_2	(10,-10)	(0,0)

ERASER [Kiekintveld et al. AAMAS'09] (MIP for multiple resources)

$$max\ d$$

$$\sum_t a_t = 1$$

$$\sum_t c_t \le m$$

$$\forall_t \qquad d - \sum_t (R_t^d c_t + R_t^u (1 - c_t)) \le (1 - a_t) M$$

$$\forall_t\ 0 \le k - \sum_t (C_t^d c_t + C_t^u (1 - c_t)) \le (1 - a_t) M$$

$$a_t \in \{0, 1\}$$

$$c_t \in [0, 1]$$

T is the set of targets
R_t is the defenders payoff given *t* (d – defended and u – undefended)
C_t is the attackers payoff given *t* (d – defended and u – undefended)
c_t is the probability that the defender defends target *t*
m is the number of defense resources the defender has
$a_t = 1$ for the target that is attacked
M is a large number

Defense at a cost [Letchford & Vorobeychik UAI'12]

- Each target *t* is assigned a cost c_t
- The defender can choose to pay c_t to prevent an attack that originates at target *t*
  - Multiple options with different efficiency may be available
  - This cost is paid even when *t* is not attacked
- Paying a fraction of c_t will offer partial protection
  - This corresponds to playing a mixed strategy
- The existing MIP can be modified to solve this problem
  - Furthermore, using techniques similar to what are used in[1] we are able to give an efficient linear programming formulation for the problem

[1] Conitzer and Sandholm 2006

Goals

- Two conflicting goals for the defender:
  - Minimize expected loss from an attack
  - Minimize amount spent on defense
  - We define an optimal solution as one that minimizes the sum of these two values

- Simple goal for the attacker:
  - Given a defense strategy for the Defender, attack the target that gives the largest expected payoff
  - In the zero-sum case, this corresponds to hurting the Defender as much as possible

Linear program for defense costs

$$\forall_{\hat{t}}\; maximize\; d^{\hat{t}} - \sum_t \sum_o p_{o,t} c^{\hat{t}}_{o,t}$$

s.t.

$$\forall_{o,t}\; c^{\hat{t}}_{o,t} \in [0,1]$$

$$\forall_t \sum_o c^{\hat{t}}_{o,t} = 1$$

$$d^{\hat{t}} - \sum_o (R_{o,\hat{t}} c^{\hat{t}}_{o,\hat{t}}) = 0$$

$$k^{\hat{t}} - \sum_o (C_{o,\hat{t}} c^{\hat{t}}_{o,\hat{t}}) = 0$$

$$\forall_t\; 0 \leq k^{\hat{t}} - \sum_o (C_{o,t} c^{\hat{t}}_{o,t})$$

T is the set of targets
O is the set of defense options
$R_{o,t}$ is the defenders payoff given *o* and *t*
$C_{o,t}$ is the attackers payoff given *o* and *t*
$p_{o,t}$ is the price (cost) of defense option *o* for target *t*
$c_{o,t}$ is the probability that the defender defends target *t* with option *o*

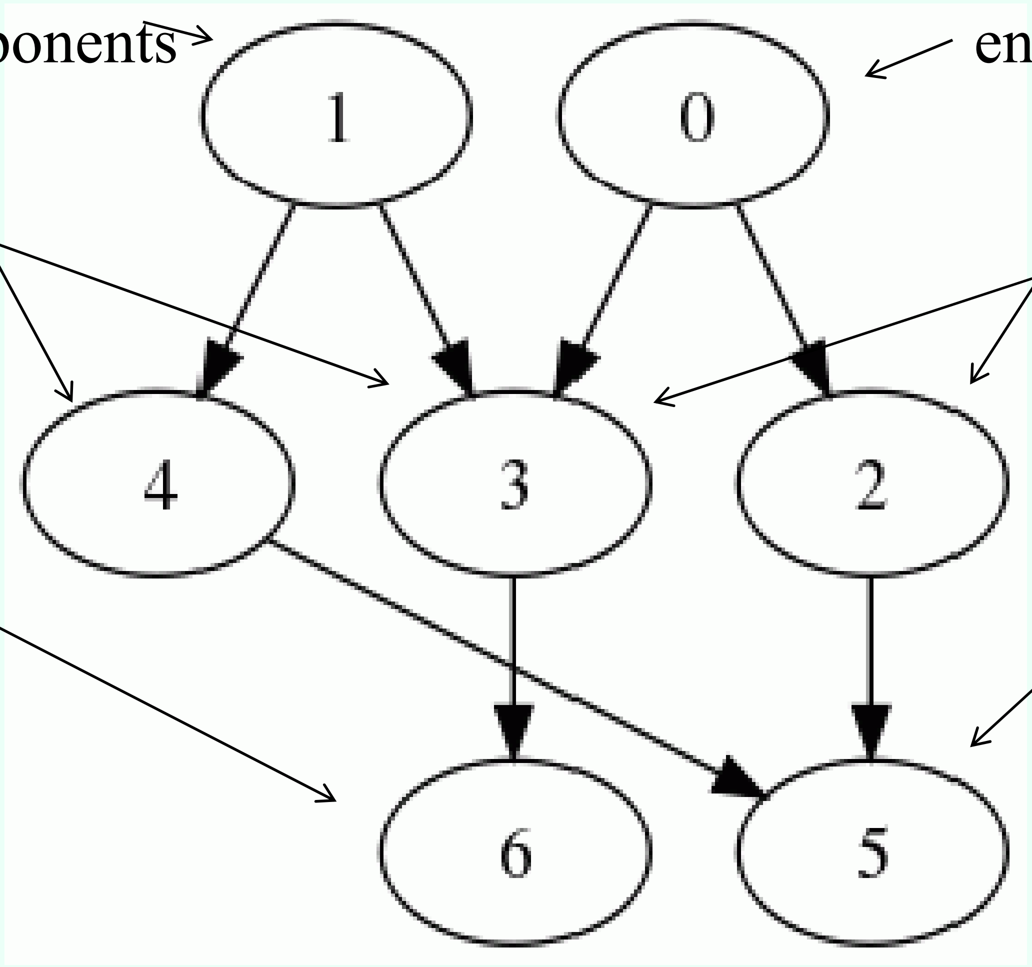
A simple example (car supply chain)

Produces electronic components

Produces engine components

Produces radios

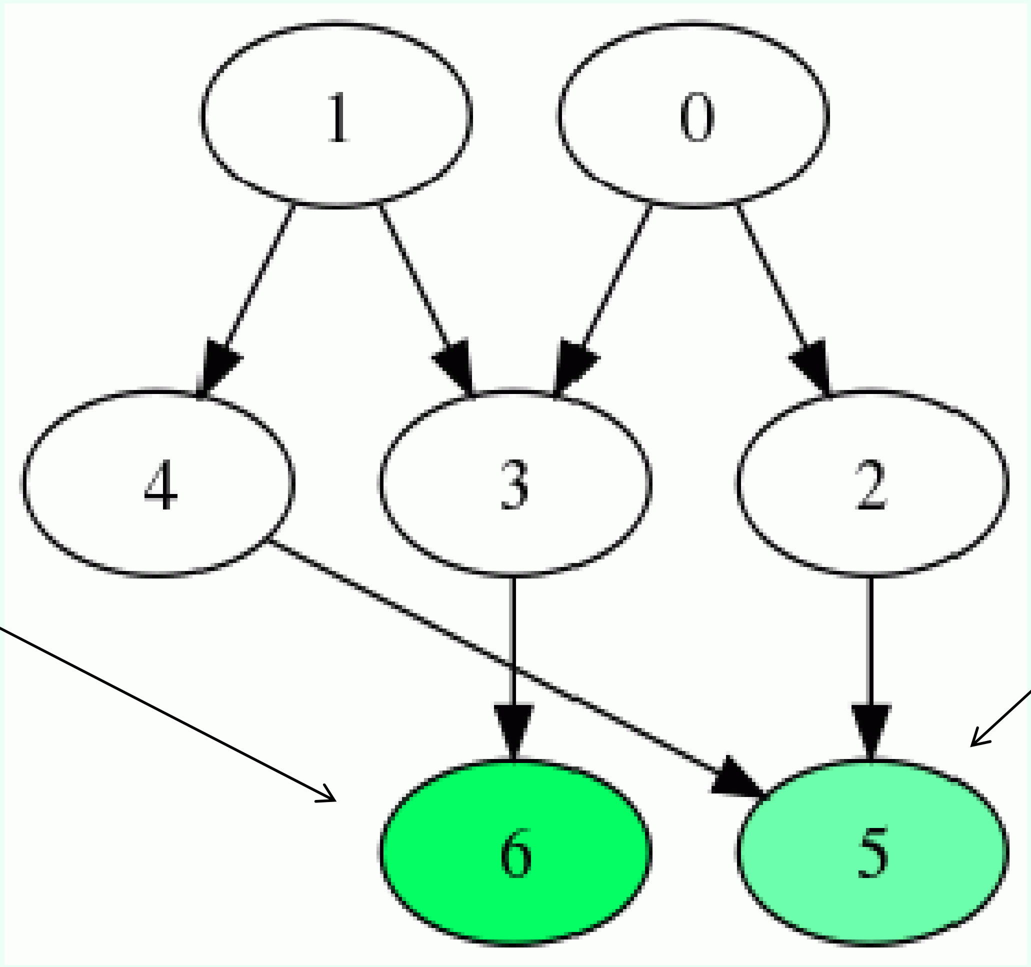
Produces engines



Produces Micro-car

Produces SUV

Valuation



$$\begin{tikzcd} 1 \arrow[d] \arrow[dr] & & 0 \arrow[dl] \arrow[d] \\ 4 \arrow[drr] & 3 \arrow[d] & 2 \arrow[d] \\ & 6 & 5 \end{tikzcd}$$

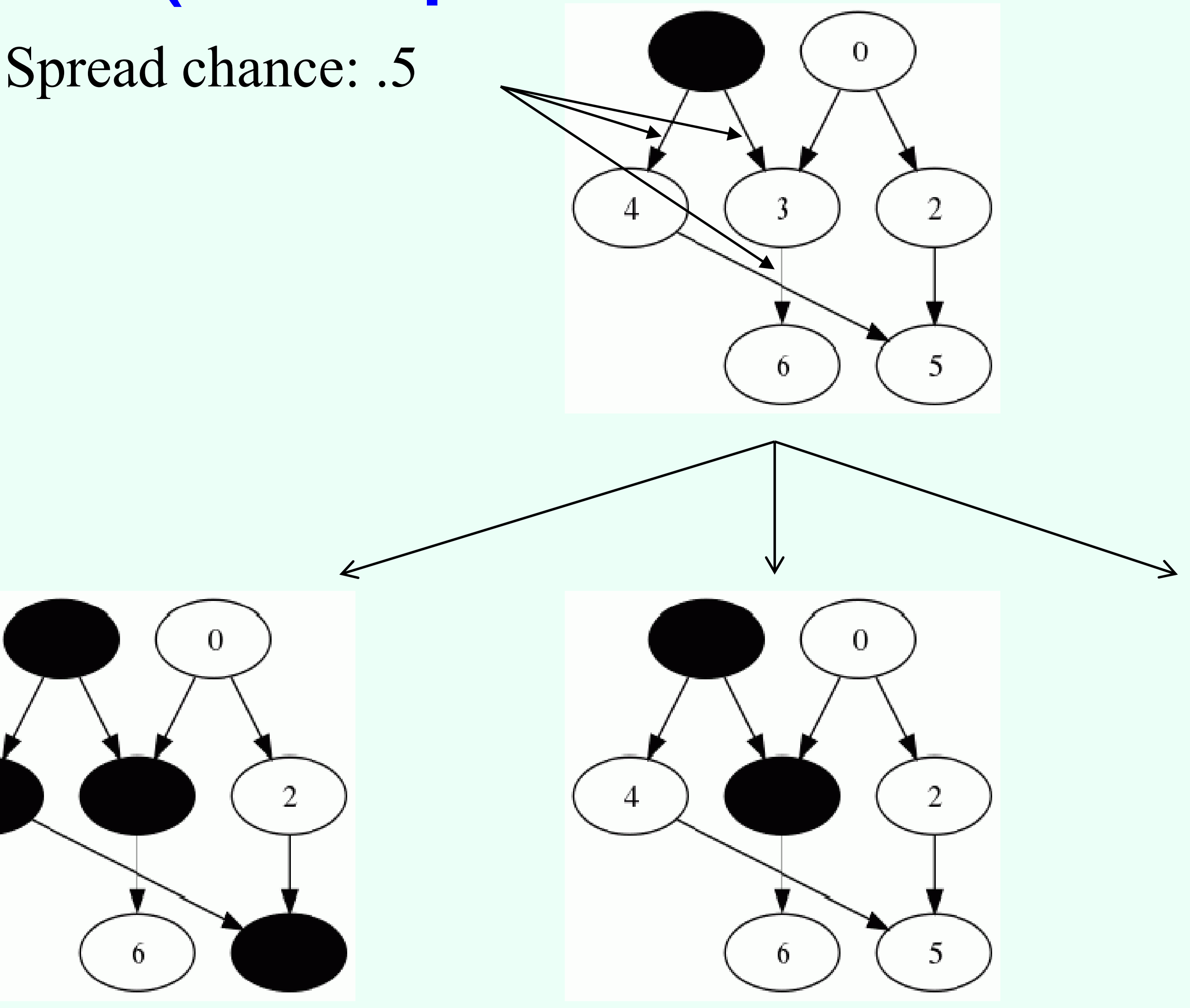
Produces
Micro-car

Value = 2.168

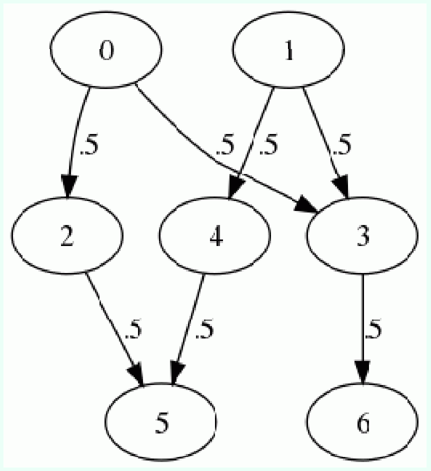
Produces
SUV

Value = 1.227

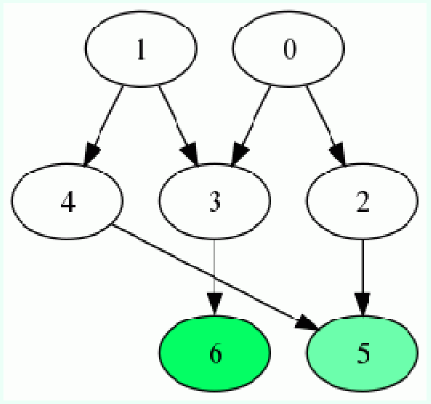
Attack model (Independent cascade)



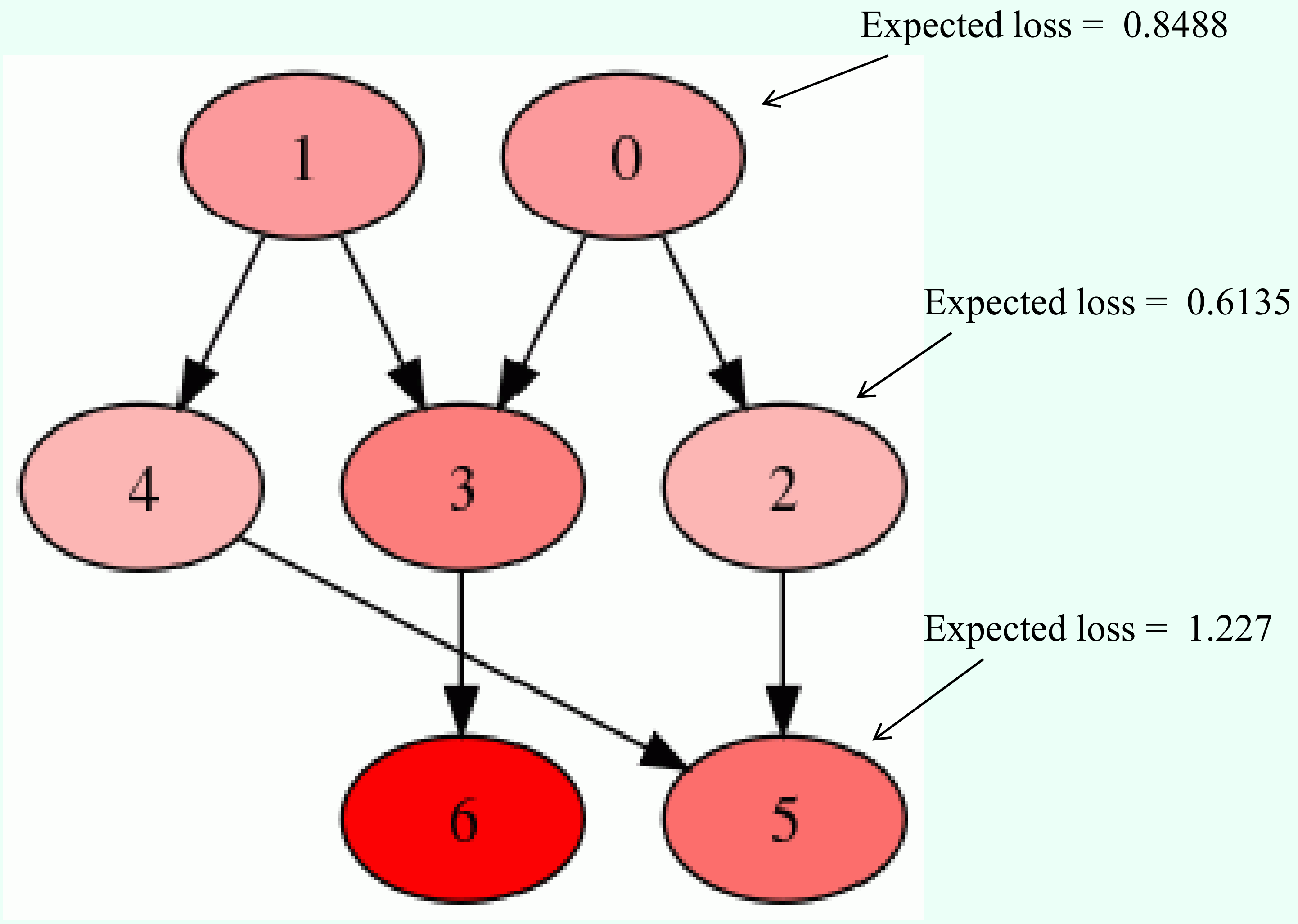
Evaluating expected loss



Cascade model



Values



Evaluating expected loss

- Related to the idea of maximizing influence
  - Which unfortunately is NP-hard[1]
- However, extremely easy to approximate through simulation[1]
- Fast algorithms for special cases
  - Two-pass algorithm for undirected trees (O(|T|))
  - Domain specific procedures (i.e. consequence analysis)

[1]Kempe et al. 2003

Notation for two-pass algorithm for undirected trees

- T : set of targets
- O : set of defense options
- $z_{o,t}$: probability of an attack succeeding at target t under defense option o
- $p_{t,t'}$: probability of a cascade from t to t'
- N_t : set of the neighbors of t
- w_t : worth (value) of target t
- P_t : Parent of t

[1]Kempe et al. 2003

Expected loss in trees

- Expected loss due to cascading failure:

$$z_{o,t} \sum_{t' \neq t} w_{t'} p(failure(t')|t)$$

- *p(failure(t')|t)* = product of probabilities of the edges on the path between t and t'
- By organizing these paths, we can express the expected loss of the contagion spreading across an edge (t,t') as:

$$E[U_{(t,t')}] = p_{t,t'} \left(w_{t'} + \sum_{t'' \in N_{t'}, t'' \neq t} E[U_{(t',t'')}] \right)$$

Utility evaluation

Given expected losses over edges, we can calculate expected losses for a target t:

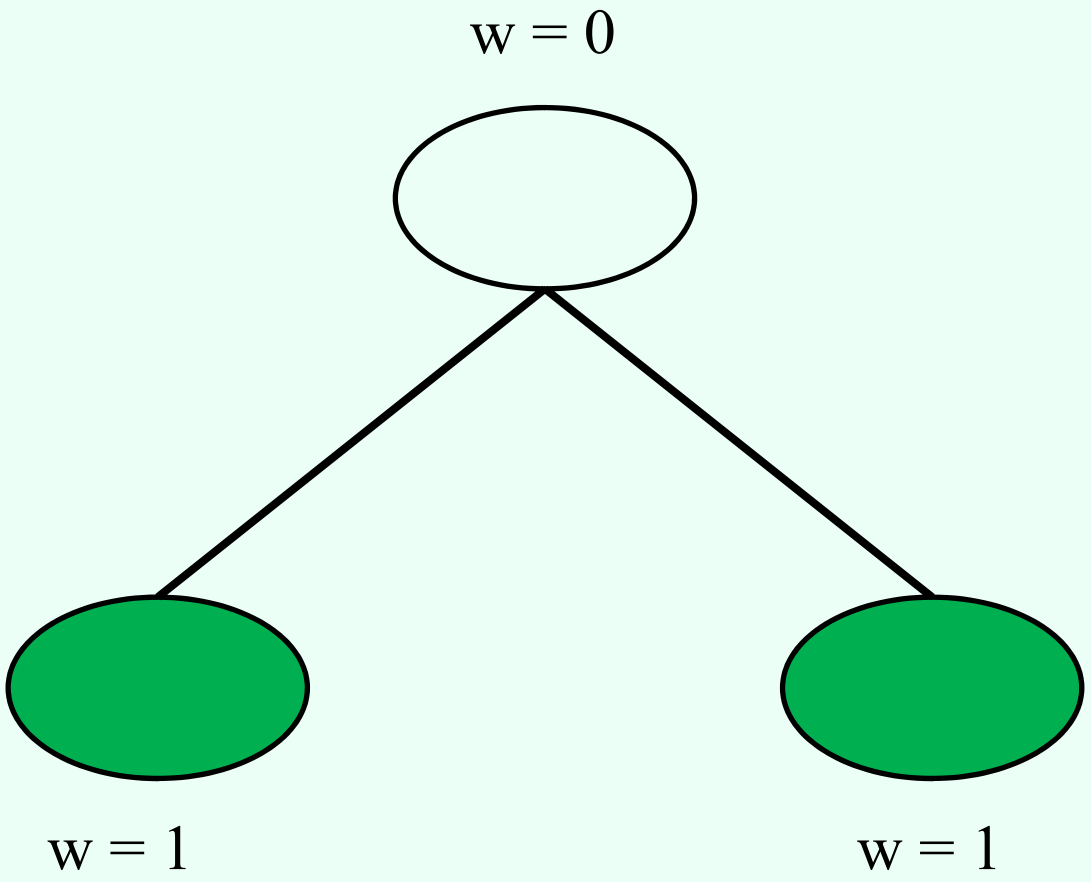
$$U_t = w_t + \sum_{t' \in N_t} E[U_{(t,t')}]$$

Two-pass algorithm for undirected trees

- Pick a random node to be the root
- Break each edge into two directed edges
- Upward pass
  - Calculate expected loss for each edge from parent to child
- Downward pass
  - Calculate expected loss for each edge from child to parent

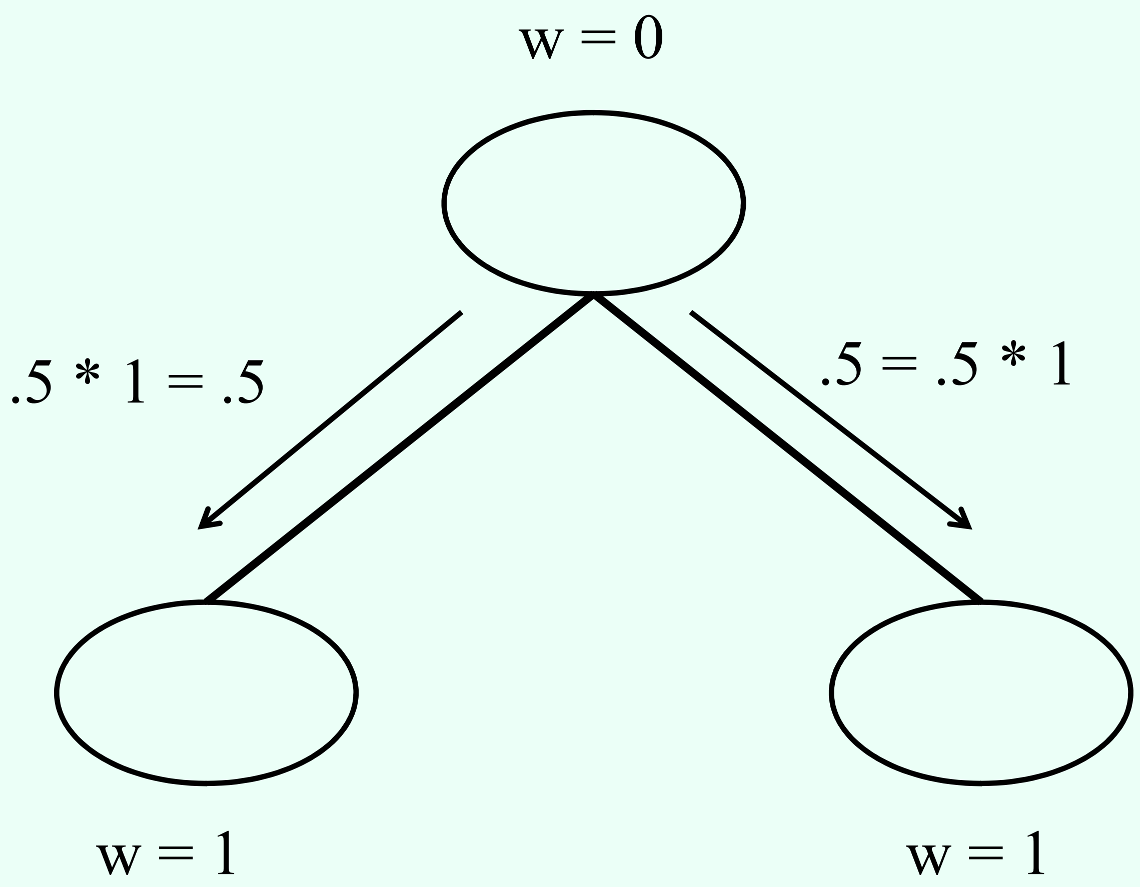
Simple example

P(spread) = .5

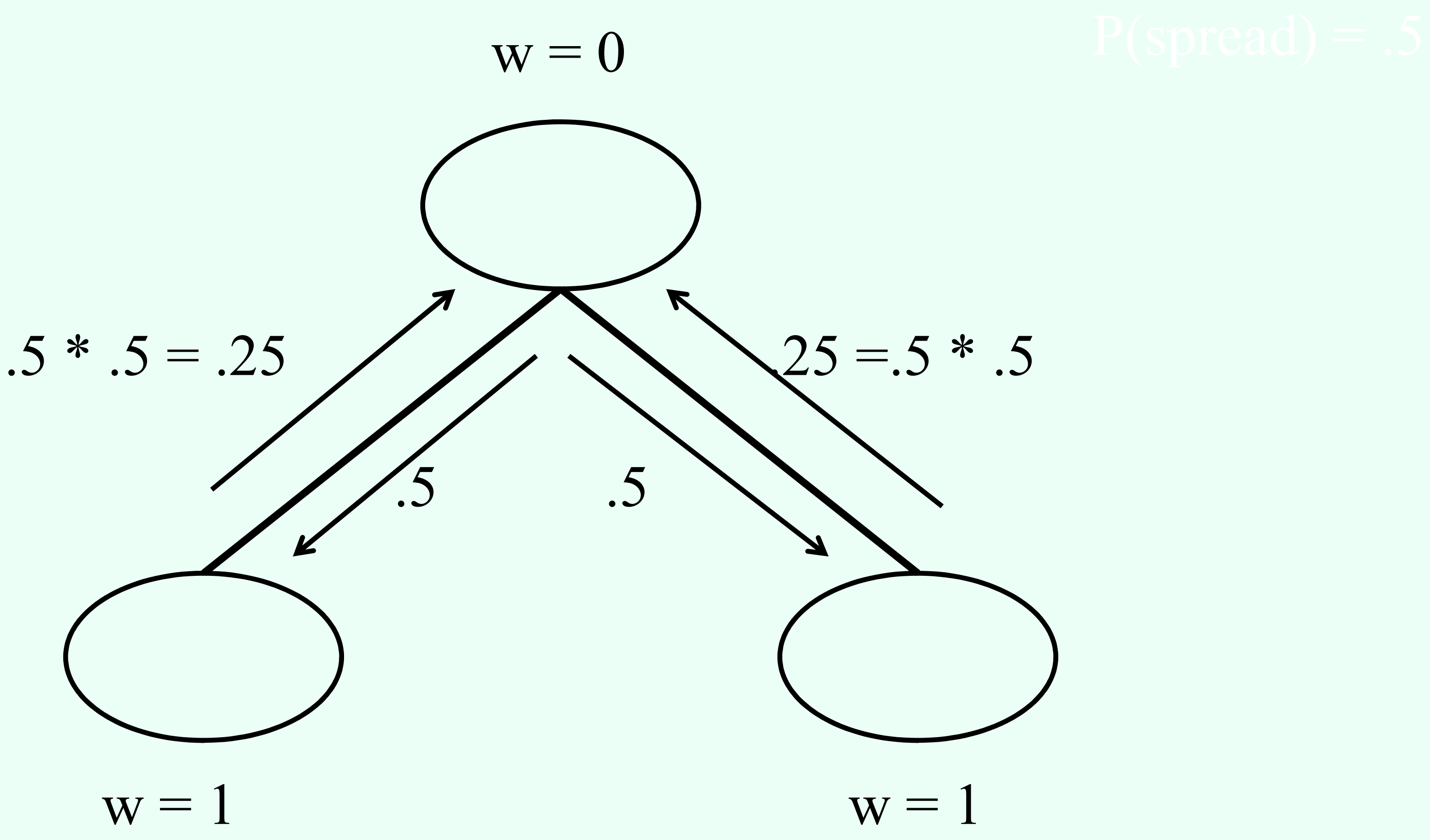


Upward pass

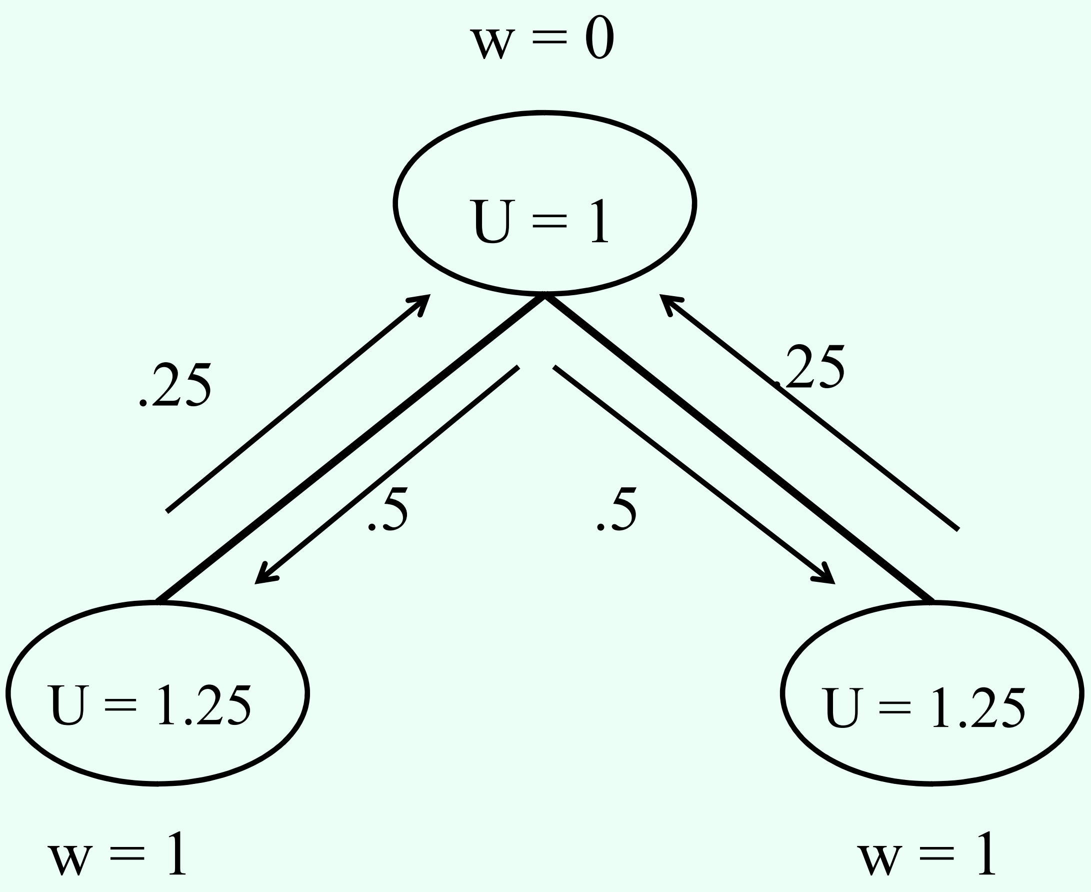
P(spread) = .5



Downward pass



Expected loss calculation



P(spread) = .5

Correctness

- We model this as a message passing algorithm
  - To calculate $E[U_{(t',t)}]$ requires the messages from $N_t \setminus t'$ to t

- Upward
  - Each node has only one parent
  - All children have previously passed messages to t
  - Thus, each node has $N_t \setminus P_t$ available when generating the message to its parent

- Downward pass
  - All children passed messages to t in the upward pass
  - Parent has already passed message in this pass
  - Thus, each node has N_t available

Achieving linear time

- As given, doesn't achieve linear time
  - A node with O(n) edges (star) requires O(n^2) edge queries
- To get around this, need to store values at the nodes
  - We can reason that:

$$U_t = w_t + \sum_{t \in N_t} p_{t,t'} (U_{t'} - E[U_{(t',t)}])$$

- Store a running total at each node
  - By the same reasoning as before the necessary calculations have been performed before they are needed as inputs
  - However, now need to show that we can recover the needed values from the stored value

Recovering the correct values

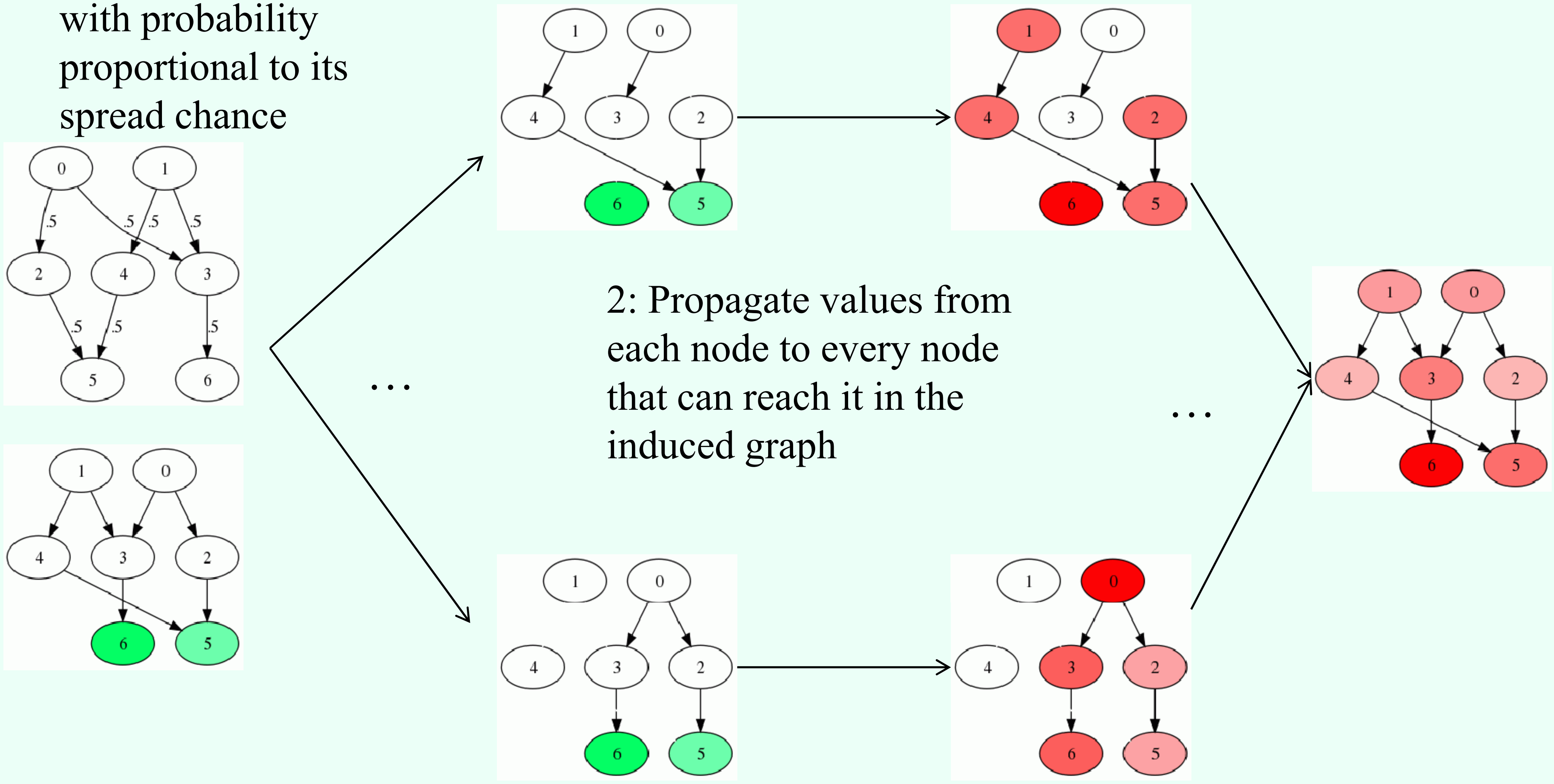
- Upward pass
  - Already processed all of the children but not the parent
  - Value stored is exactly what is needed
- Downward pass
  - Value stored is not correct
  - Children have not been updated yet
  - Can subtract out the value stored at each child to recover the needed values

Runtime

- Visit each edge twice
  - Once on upward and once on downward pass
- Perform a constant amount of work each time
  - Need to query the source of the edge
  - On downward pass also need to query the target of the edge
  - Also need to update the target of the edge
- Since this is a tree, |T| - 1 = |E|
  - Thus, runtime is O(|T|)

Approximation through simulation

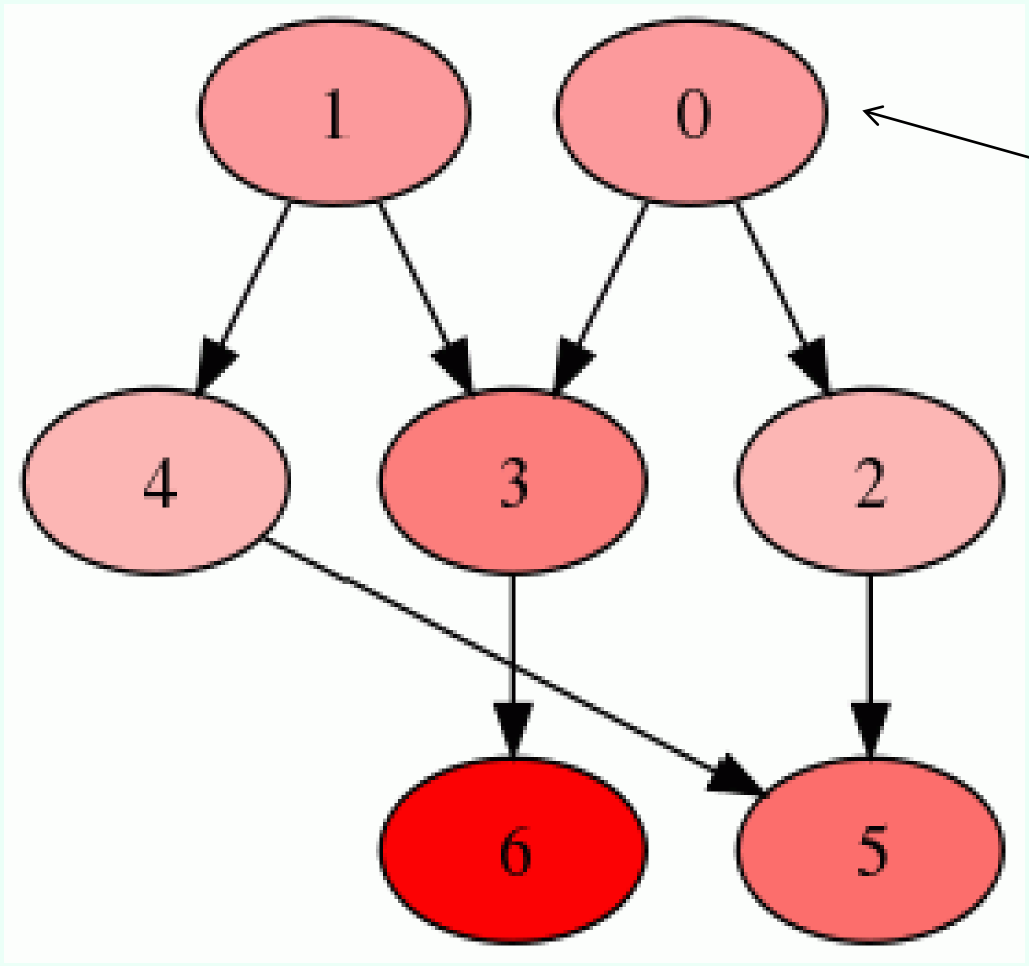
1: Take each edge with probability proportional to its spread chance



2: Propagate values from each node to every node that can reach it in the induced graph

3: Take the average

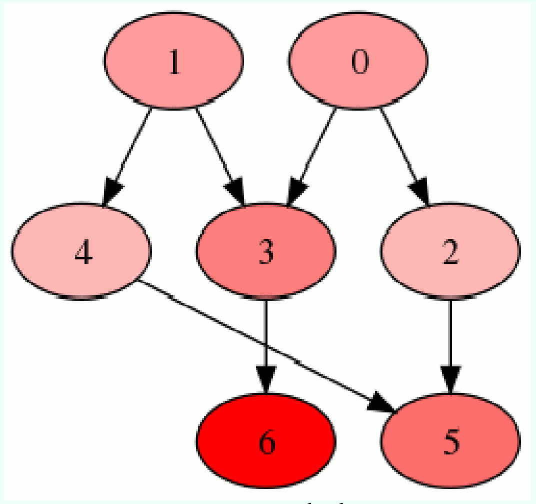
Formulating this as a game (zero sum)



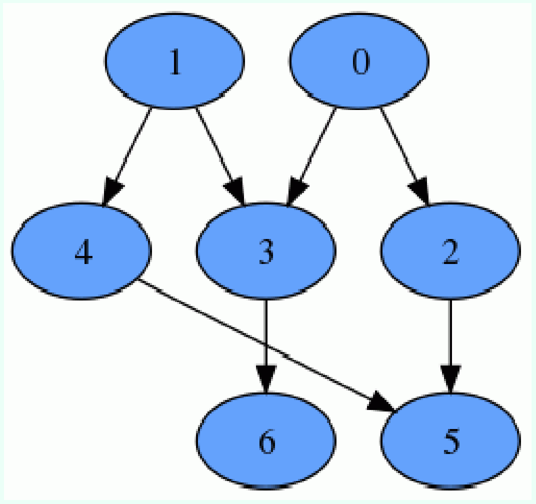
If target **0** is attacked

Defended	(0,0)
Not Defended	(-0.8488,0.8488)

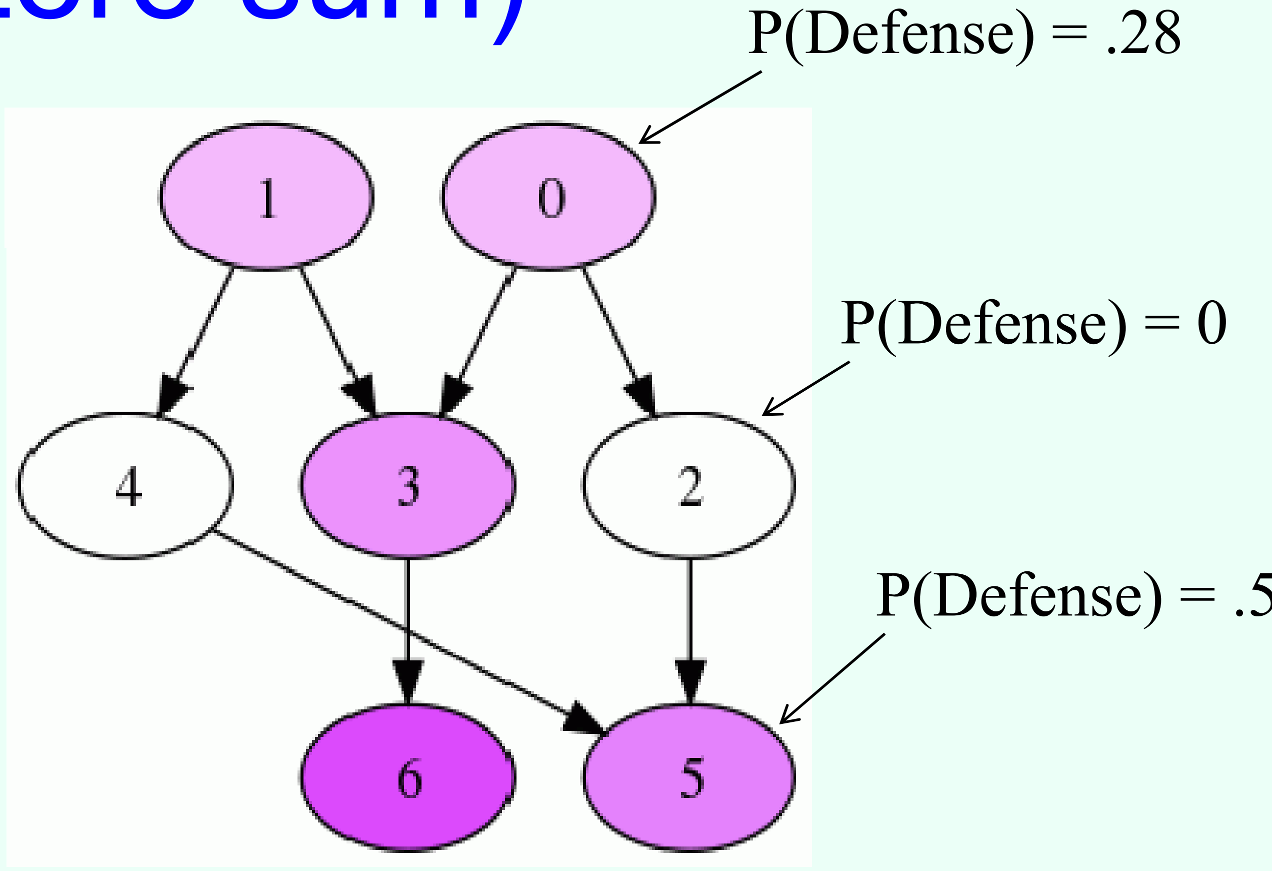
Optimal defense strategy (zero sum)



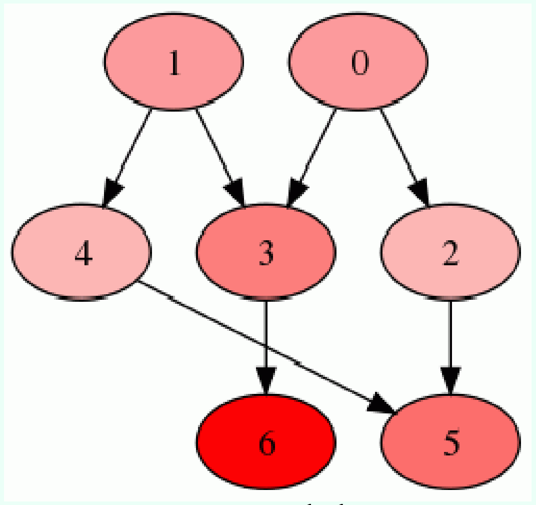
Expected loss



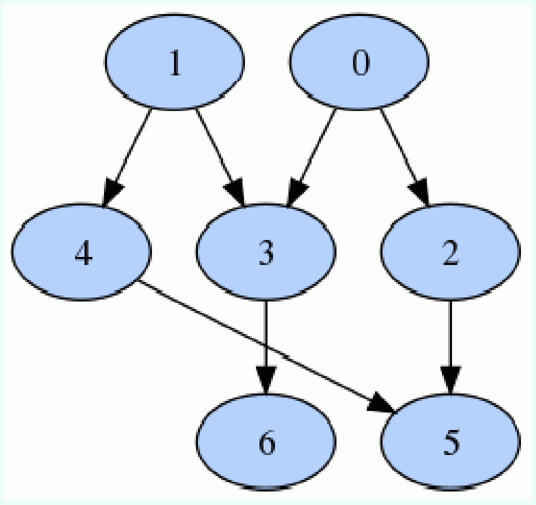
Uniform cost of .1428



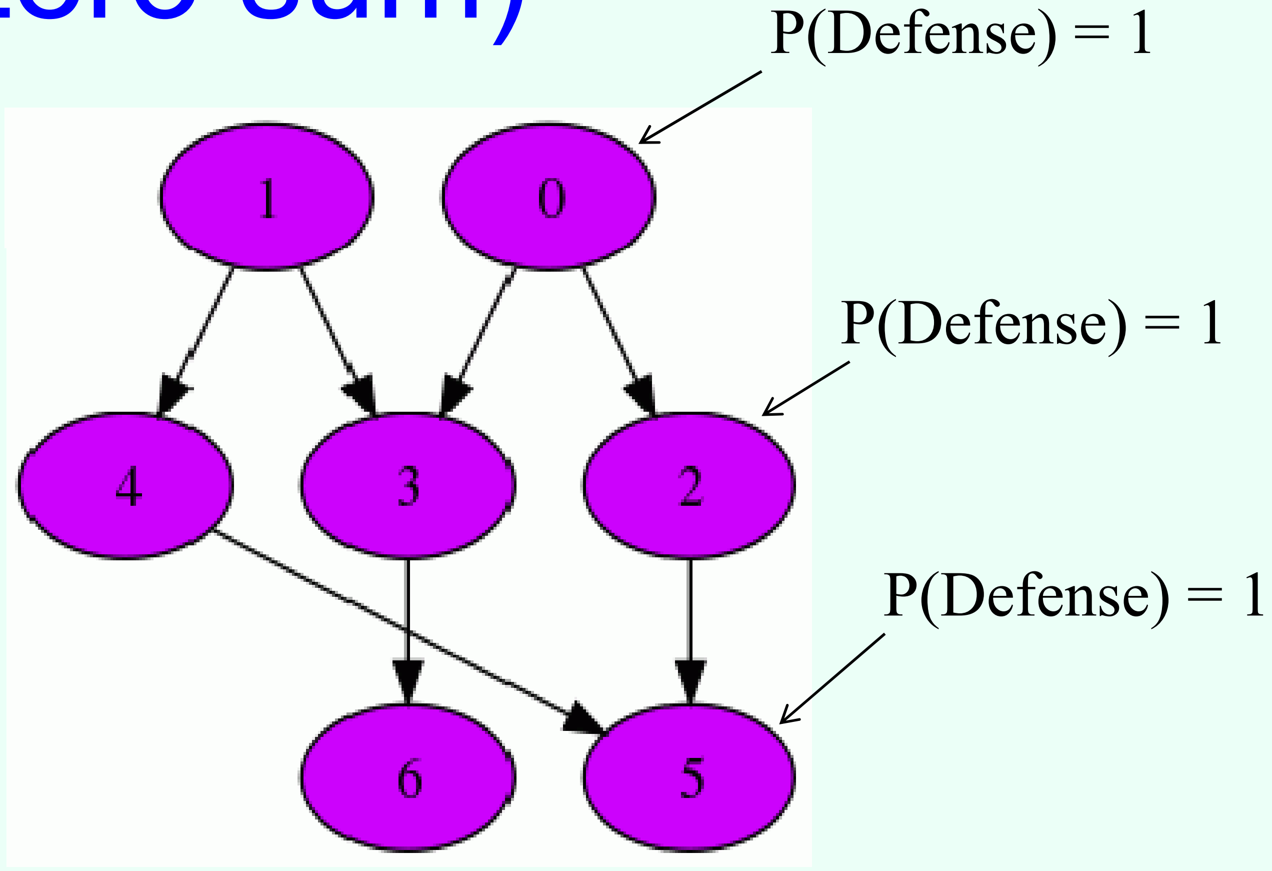
Optimal defense strategy (zero sum)



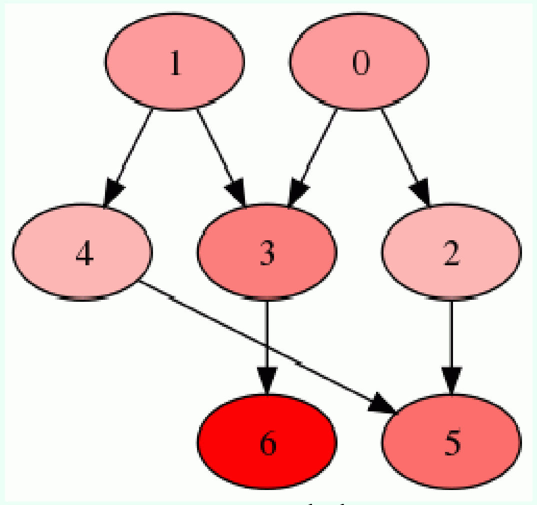
Expected loss



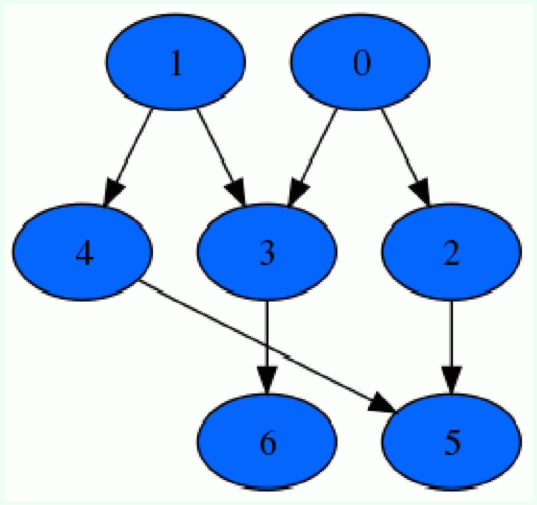
Uniform cost of .0179



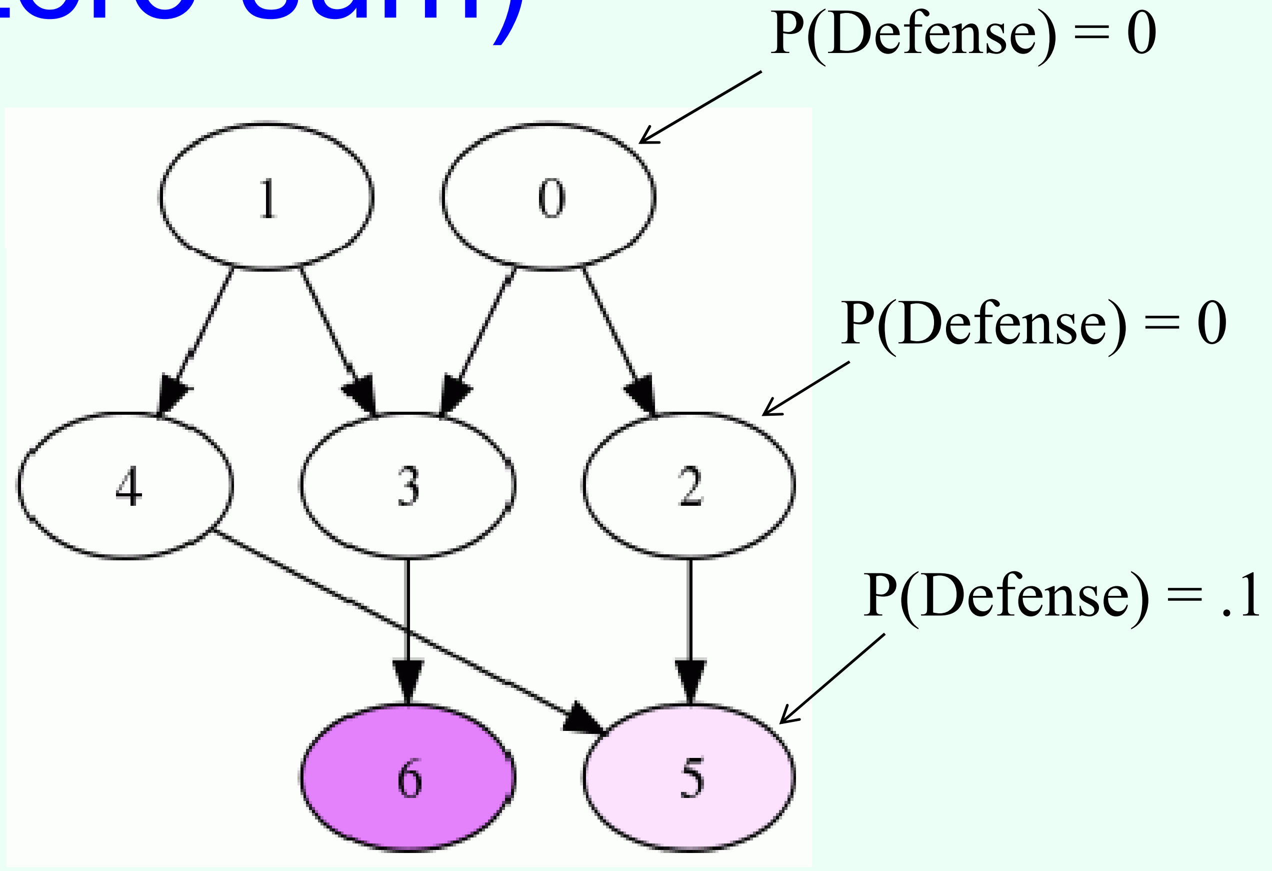
Optimal defense strategy (zero sum)



Expected loss



Uniform cost of .5714


P(Defense) = 0
P(Defense) = 0
P(Defense) = .1
1
0
4
3
2
6
5