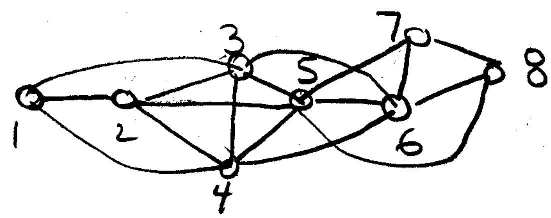
Computational Game Theory (CIS 620/OPIM 952)
Instructor: M. Kearns
Luis Ortiz   February 4, 2003

Graphical Games: Part 4

Relating NashProp to Constraint Propagation Algs. for Generalized Arc Consistency

Arc consistency in constraint networks

- Let E be edges in the constraint network:
- Ex:



- For CSPs with binary constraints (i.e., functions of 2 variables)

Typically only interested in maximal sub-domains D'_j

- An arc $(i,j) \in E$ is consistent wrt $D'_i \subset D_i, D'_j \subset D_j$ if
  $\forall p_i \in D'_i, \exists p_j \in D'_j$ s.t. $P_i = p_i, P_j = p_j$ is consistent [with their constraint]
  $[C_{(P_i,P_j)}(p_i,p_j) = 1]$.
- A constraint network is arc-consistent [wrt $(D'_1, \dots, D'_n)$] if all arcs (i,j) are consistent [wrt (D'_i, D'_j)].

- An alg. for computing arc-consistency

Initialize: $\forall i = 1, \dots, n$, $\forall p_i \in D_i$, $T_{P_i}(p_i) = 1$!

One function for each variable.

Iterate: for every arc (i,j) in the constraint net (in some order!),

for all $p_i \in D_i$, $T_{P_i}(p_i) = 1$ iff "P_i believes there exist a consistent assignment where $P_i = p_i$"

iff $\exists p_j \in D_j$ s.t.

1. $T_{P_j}(p_j) = 1$
2. $P_i = p_i$ and $P_j = p_j$ consistent with arc constraint $[C_{(P_i,P_j)}(p_i,p_j) = 1]$.

[Note similarity with Table updates in TreeNash & Nashprop.]

To note alg. correctness, let $D_i' = \{ p_i \in D_i : T_{P_i}(p_i) = 1 \}$ $\forall i = 1, \ldots, n$.
(once, alg. terminates)

To note that alg. terminates, assuming the domains D_i are finite, note that at every "round" (consideration of every arc), either

ⓐ At least one entry in at least one "table" $T_{P_i}(p_i)$ changed from 1 to 0.
[the corresponding value assignment for the corresponding variable was found inconsistent with <u>some</u> constraint].

or ⓑ No entry changed $\Rightarrow$ convergence.

So if d = max. domain size, e = # of arcs $(\equiv |E|)$, running time
$O(e\, d \cdot d^2) = O(e\, d^3)$

[Proof idea just as for NashProp].

(Generalized) (GAC)

Arc consistency in constraint networks (CN)

- For binary constraints (functions of 2 vars), each arc (edge) in the CN corresponds exactly to a single constraint

- For non-binary constraints (functions of more than 2 vars), this no longer holds.

- Notion of (generalized) arc consistency for CNs with non-binary constraints is defined over constraints:

Defn: A constraint C_ℓ is arc-consistent w.r.t. domains $\{D_j' \subset D_j\}_j$ of the vars $\{P_j\}_j$ in the constraint if

1. for every variable P_i in the constraint C_ℓ, and for every value $p_i \in D_i'$,
   there exists values $\vec{p}_{\setminus i} \in \times_{\substack{j \neq i \\ P_j \text{ in } C_\ell}} D_j'$ for the variable $\vec{P}_{\setminus i}$ which are vars. in C_ℓ other than P_i,
   s.t.
   $P_i = p_i$ and $\vec{P}_{\setminus i} = \vec{p}_{\setminus i}$ is consistent with C_ℓ.
2. each D_j' is largest possible satisfying condition 1.

Defn: A CN is arc-consistent w.r.t. $\{D_i'\}$ if each constraint C_ℓ is consistent wrt $\{D_j' : P_j \text{ in } C_\ell\}$.

An alg. for GAC in CNs.

Intuitive interpretation of $T_{P_i}(p_i)$:
$T_{P_i}(p_i)=1$ iff we believe there exist some satisfying assignment for all vars with $P_i = p_i$

Initialization: $\forall$ vars P_i, $\forall$ values $p_i \in D_i$, $T_{P_i}(p_i)=1$

Run until no change in $\{T_{P_i}\}$

$\forall$ constraint C_ℓ (in some order)

$\forall$ variable P_i in C_i (in some order),

$\forall$ value $p_i \in D_i$,

$T_{P_i}(p_i)=1$ iff $\exists$ values $\vec{p}_{\setminus i} \in \times_{j \neq i;\, P_j \text{ in } C_\ell} D_j$ for the variables $\vec{P}_{\setminus i}$ in C_ℓ other than P_i, s.t.

1. $T_{P_j}(p_j)=1$, $\forall j \neq i : P_j$ in C_ℓ
2. $P_i = p_i$, $\vec{P}_{\setminus i} = \vec{p}_{\setminus i}$ <u>consistent</u> with C_ℓ.

- $D_i' \equiv \{p_i \in D_i : T_{P_i}(p_i)=1\}$
- Easy to see correctness: Alg. is just "running the definition," and removing values from the original domain of each variable (i.e., $T_{P_i}(p_i)=0$) if value found "inconsistent" with any satisfying assignment for all the vars.
- Convergence also easy to see: A "contraction"
- If finite domains, finite (poly.) running time easy to see.

- Note similarity of GAC alg. and NashProp.

- Need to introduce other concepts/notation to establish <u>direct</u> relationship...

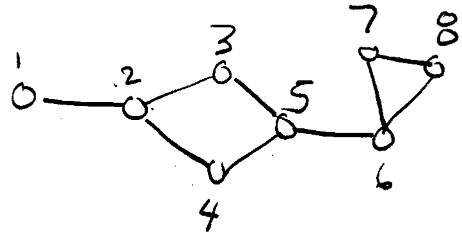
- NashProp can be seen as a particular instantiation of a constraint propagation algorithm for (generalized) arc-consistency in a (directed) constraint network resulting from a particular CSP formulation, followed by a local search with backtraking.

[This is where we are heading in the next slides...]

Directed constraint network (DCN)

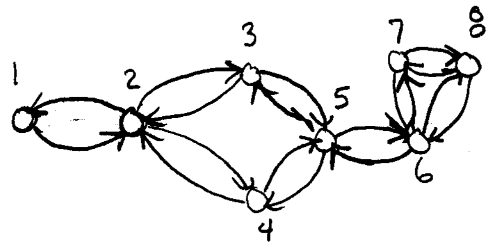
- Defined exclusively to establish exact NashProp-GAC Alg relationship
- Each variable P_i has exactly one constraint C_i associated to it.

For instance, for graphical game: [Have as many constraints as vars.]



we can associate variable P_i (player i) with constraint C_i (best-response constraint for player i).

- Each constraint represented as directed arcs to its associated variable (node) from the other vars of which the constraint is a function.



[Aside: In the context of games, DCN ≡ directed Gf.]

"Arc-consistency" (AC) in DCNs

Dfn: A constraint C_i in a DCN is arc-consistent with respect to $D_j' \subset D_j$, $\forall j$ s.t. P_j is variable in the constraint if ① for P_i the associated variable of the constraint, and $\vec{P}_{\setminus i}$ all other variables in the constraint, (parents of node i in DCN).

$$\forall p_i \in D_i', \exists \vec{p}_{\setminus i} \in \times_{j \neq i} D_j' \text{ (assignments to the other variables in the constraint)}$$

possible assignment to P_i

s.t. $P_i = p_i$, $\vec{P}_{\setminus i} = \vec{p}_{\setminus i}$ is consistent with the constraint C_i.

$$\left[\text{i.e., } C_i(p_i, \vec{p}_{\setminus i}) = 1\right]$$

and ② each D_j' is largest possible set satisfying above condition! [wrt $D_1', D_2', \ldots D_n'$]

Dfn: A DCN is arc-consistent if all its constraints C_i are arc-consistent [w.r.t. $\{D_j' : P_j$ a variable in constraint $C_i\}$]

An Alg. for AC in DCNs:

$\forall$ every node i in the DCN

$$\forall p_i \in D_i, \; T_{P_i}(p_i) = 1 \text{ iff } \exists \vec{p}_{\setminus i} \in \times_{\substack{j:\\ P_j \text{ in } C_i\\ j \neq i}} D_j$$

(assignments to the parents of P_i in DCN --- are vars in C_i)

s.t.

1. $T_{P_j}(\vec{p}_{\setminus i}[P_j]) = 1 \quad \forall j \neq i$, s.t. P_j in C_i
2. $P_i = p_i$, $\vec{P}_{\setminus i} = \vec{p}_{\setminus i}$ consistent with C_i $\left[C_i(p_i, \vec{p}_{\setminus i}) = 1\right]$.

Note: For DCN resulting from simple formulation of GG (i.e., one variable per player), running the AC above can lead to very "weak" results;

For instance, if for every player i, there exist ~~ean~~ assignment to its neighboring players which make player i indifferent then the resulting sets of subdomains for which the DCN is arc-consistent is the same as the original domains; that is

$$\forall i, \quad D_i' = D_i \; !$$

- This motivates alt. formulations in which we form vars. by merging players...

An alternative (CSP) formulation for a GG.

Let for every edge (W,V) in the graphical game

① a variable

$$P_{(W,V)}$$

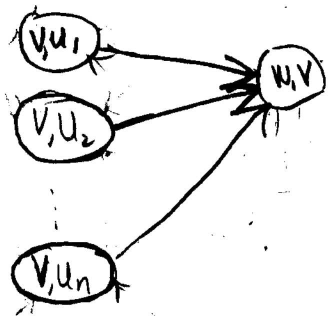
[Note: $P_{(V,W)} \neq P_{(W,V)}$!]

② a domain $D_{(W,V)}$ of possible values for $P_{(W,V)}$ s.t. it is the crossproduct of space of mixed strategies for V and W.

③ A constraint $C_{(W,V)}$ which is function of $P_{(W,V)}$ and $P_{(V,U_i)}$ where $\vec{U}$ are the neighbors of V other than W in the game graph.

and to be equivalent to the (ϵ)-best-response condition for player V.

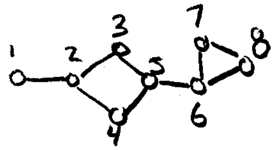
- Define a $CSP = (\{P_{(W,V)}\}, \{D_{(W,V)}\}, \{C_{(W,V)}\})$.
- For each (W,V) associate $P_{(W,V)}$ to $C_{(W,V)}$
- Graphical representation as a DCN: For each (W,V), locally, it looks like:...



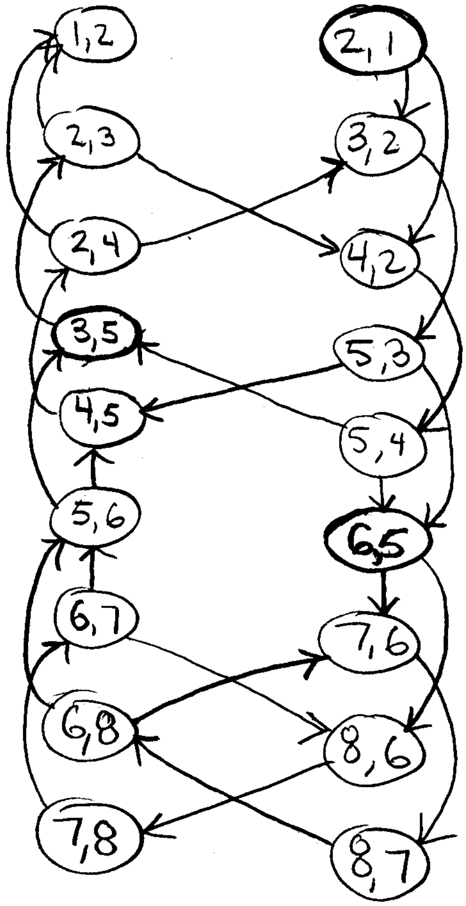
(Similar for other nodes...)

Alternative CSP formulation

Example: GG =

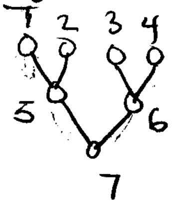


DCN for new formulation:

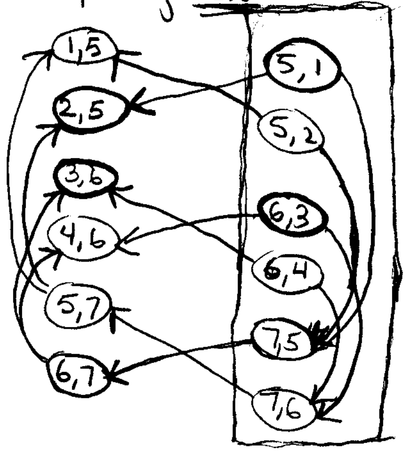


Remark: for tree GGs, DCN corresponding with this CSP formulation is acyclic!

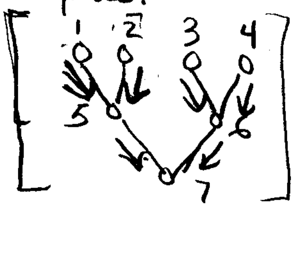
Ex: GG



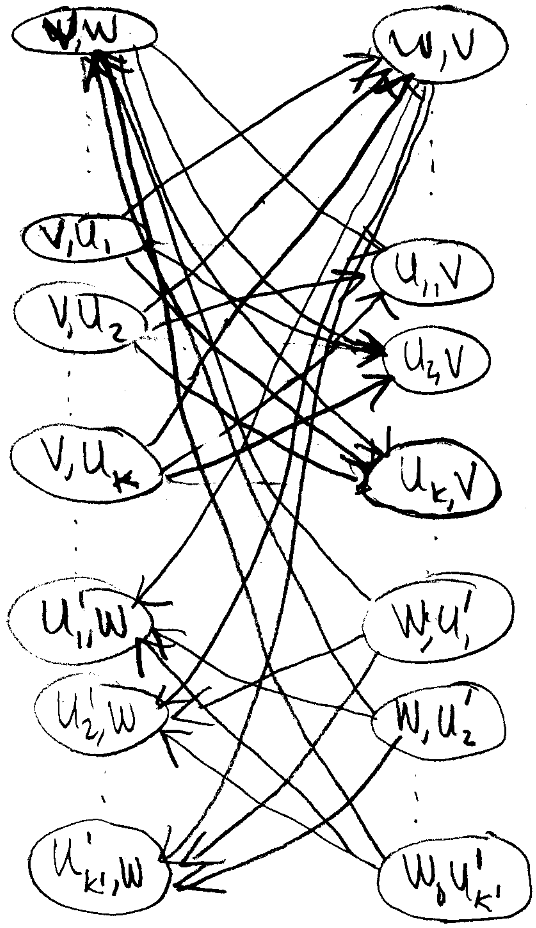
Corresponding DCN:



→ "Characterizes" TreeNash "Downstream" pass.



In general,



[$\vec{U'}$ are neighbors of W other than V]

Finally,

- Table-passing phase of Nash prop equivalent to running AC for DCNs in that formulation!

[Intuitively, Variable $I_{(w,v)}$ are the variables of which table T_{WV} [NashProp's limit tables] (sent from player V to neighboring player W, in NashProp) is a function]

More specifically,

If DCN is arc-consistent wrt $\{D'_{(w,v)}\}$ then

[$\forall$ player pairs (W,V), $T_{WV}(w,v)=1 \iff (w,v) \in D'_{(w,v)}$.]

NashProp's limit tables.

Remarks:

- More sophisticated/stronger notions of consistency exist (i.e., k-consistency)

  [Arc consistency corresponds to 2-consistency]

- However, they require more space/time for computation as strength increases.
- AC Algorithms (and in particular, NashProp) seem to strike the right balance between strength and computation difficulty.