NON SERIAL DYNAMIC PROGRAMMING

for graphical models.

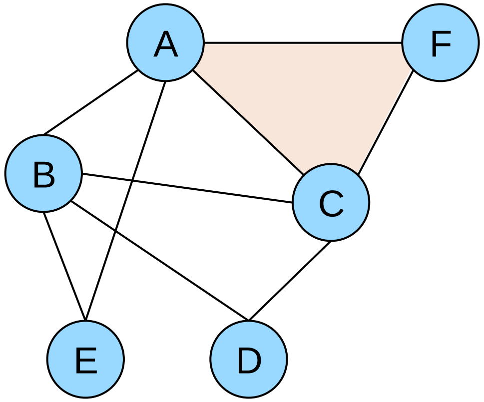
T. Schiex, R. Sabbadin

Graphical models (CSP, MRF, Bayes nets…)

- A set of variables $X = \{X_1, \ldots, X_n\}$
- A set of domains $D = \{D_1, \ldots, D_n\}$
- A set of functions Q
  - Each function $P_S \in Q$ involves a set of variables $S \subset X$

$$P_{AFC} := (2F^2 + 7AFC)$$

- Primal graph (interaction graph)
  - One vertex per variable
  - Edge (A,B) if $P_S \in Q$ s.t. $\{A,B\} \subset S$



Functions, combination

- P_S maps tuples of values of S to 1element of E
  - E= {0,1} relation (CSP, SAT)
  - E= R^+ *energy* or potentials (MRF)
  - E= [0,1] probabilities (BN), fuzzy m.degree
- Function combination by
  - Logical and relations (CSP, SAT)
  - +, x energy, potentials (MRF)
  - x, min probabilities (BN), fuzzy m.degree

A graphical model defines a joint function over X

Functions definition

- Functions can be defined by:
  - Tables (discrete domains)

A	C	F
red	green	blue
blue	red	red
blue	blue	green
green	red	blue

A	C	F	P(F|A,C)
0	0	0	**0.14**
0	0	1	**0.96**
0	1	0	**0.40**
0	1	1	**0.60**
1	0	0	**0.35**
1	0	1	**0.65**
1	1	0	**0.72**
1	1	1	**0.68**

  - Analytic formulae
    - General form

$$P_{ACF} := (F = A + C)$$

    - Pseudo-boolean polynomials, weighted clauses (boolean domains)

$$P_{AFC} := (5AFC + 3(1 - A)C)$$

$$P_{AFC} := \{(\neg A \vee \neg F \vee \neg C, 5), (A \vee \neg C, 3)\}$$

  - Arbitrary computer code

Marginalization, elimination

- To extract synthetic information on the joint

$\Sigma_X \Pi_i P_i$
- Sum, all variables: Z, #SAT, #CSP…

$\Sigma_{X-\{A\}} \Pi_i P_i$
- Sum, all but one variables: Marginal probabilities

$\vee_X \wedge_i P_i$
- Logical or (relations): CSP, SAT

$\max_X \Pi_i P_i$
- Min/Max : MAP (MRF), MPE (BN)
- Weighted CSP/SAT, Fuzzy CSP…

Idempotent elimination: NP-complete decision problems

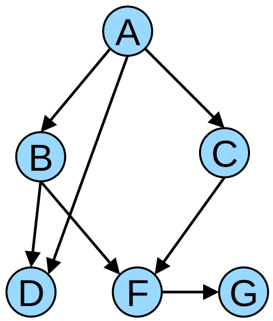
Non idempotent: #P-complete problems

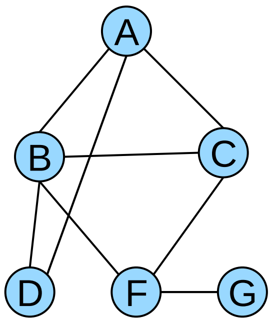
Various algorithmic approaches

- Conditioning and backtrack search exact
- Stochastic optimization/Sampling approx
- …
- **Non serial dynamic programming** exact
  - **Variable elimination**
  - Block by block elimination
  - Cluster/Join tree elimination – message passing

Non Serial Dynamic Programming

- Exploits scopes and distributivity recursively





$$P(a, g = 1) = \sum_{c,b,f,d,g=1} P(g|f)P(f|b,c)P(d|a,b)P(c|a)P(b|a)P(a)$$

Symbolic computation (distributivity)

$$P(a, g=1) = \sum_{c,b,f,d,g=1} P(g|f)P(f|b,c)P(d|a,b)P(c|a)P(b|a)P(a)$$

$$= P(a)\sum_c P(c|a)\sum_b P(b|a)\sum_f P(f|b,c)\sum_d P(d|b,a)\sum_{g=1} P(g|f)$$

$\lambda_G(f) = \sum_{g=1} P(g|f)$

$$P(a)\sum_c P(c|a)\sum_b P(b|a)\sum_f P(f|b,c)\lambda_G(f)\sum_d P(d|b,a)$$

$\lambda_D(a,b) = \sum_d P(d|a,b)$

$$P(a)\sum_c P(c|a)\sum_b P(b|a)\lambda_D(a,b)\sum_f P(f|b,c)\lambda_G(f)$$

$\lambda_F(b,c) = \sum_f P(f|b,c)\lambda_G(f)$

$$P(a)\sum_c P(c|a)\sum_b P(b|a)\lambda_D(a,b)\lambda_F(b,c)$$

$$P(a)\sum_c P(c|a)\lambda_B(a,c)$$

$$P(a)\lambda_C(a)$$

- At most 3 variables involved
- Cubic time/space only

INRA

Main properties

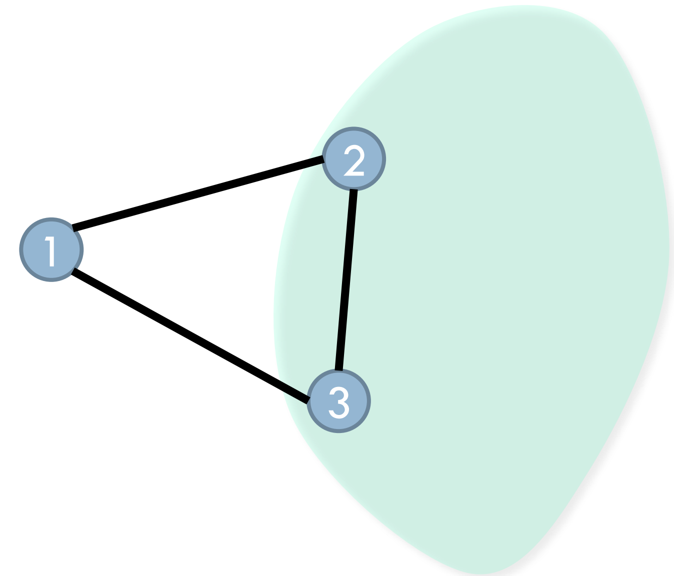
- Replacing two functions by their combination preserves the problem

- If f is the only function involving variable A, replacing f by the function
$$\lambda_A(\ldots) = \sum_a f(A, \ldots.)$$
preserves the marginal

Variable elimination

- X_i , a variable
- F_i , cost functions involving it
- O_i , other variables in F_i

$$\lambda_{X_i}(\ldots) = \sum_{a \in D_i} \prod_{f \in F_i} f(a, \ldots)$$



- Eliminate X_i and F_i
- Add λ_{X_i} to F

- One less variable
- Less cost functions
- Same marginal

Variable elimination

- Fix a variable ordering
- Successively eliminate variables
- The function at the end is your result

- The complexity depends on
  - The primal graph (interaction) structure
  - The variable ordering chosen
  - Space/time exponential in the largest # of neighbors

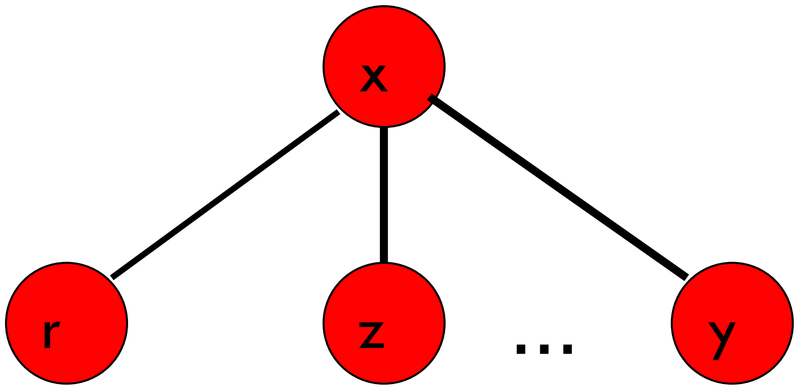
For a fixed ordering

- Play the « elimination game » on the primal graph
  - Remove variable
  - Do not compute extra functions, just add edges
  - Repeat…
  - Remember the largest # of forward neighbors

This number is called the induced width of the ordering

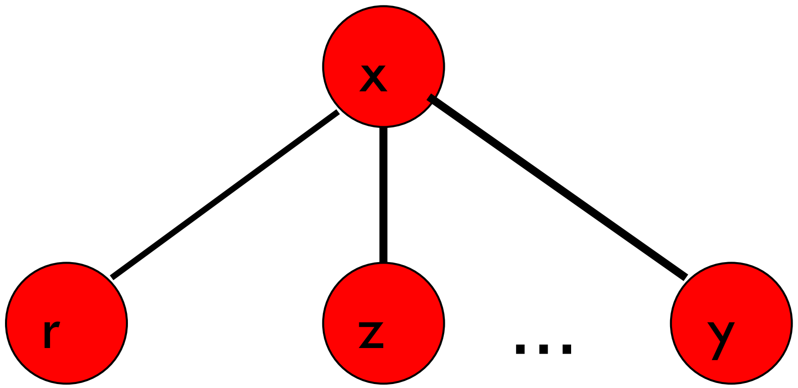
Elimination order influence

- {f(x,r), f(x,z), …, f(x,y)}
- Order: r, z, …, y, x



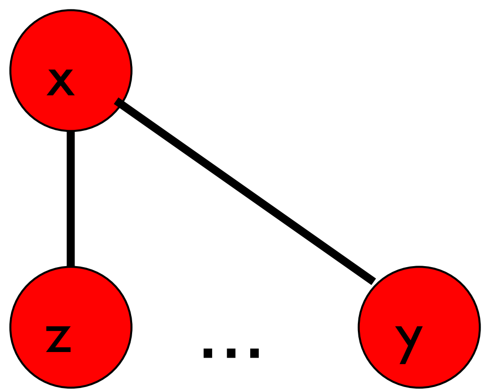
Elimination order influence

- {f(x,r), f(x,z), …, f(x,y)}
- Order: r, z, …, y, x



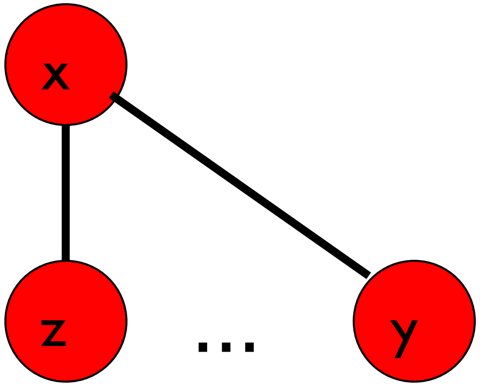
Elimination order influence

- {f(x), f(x,z), …, f(x,y)}
- Order: z, …, y, x



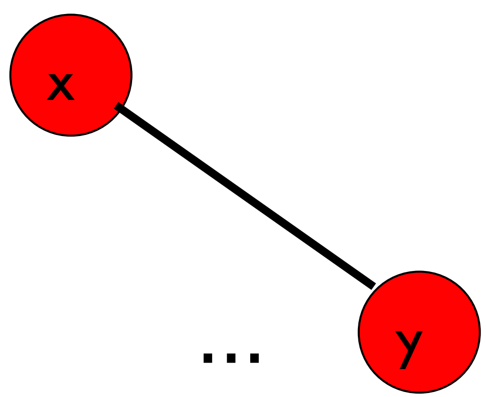
Elimination order influence

- {f(x), f(x,z), …, f(x,y)}
- Order: z, …, y, x



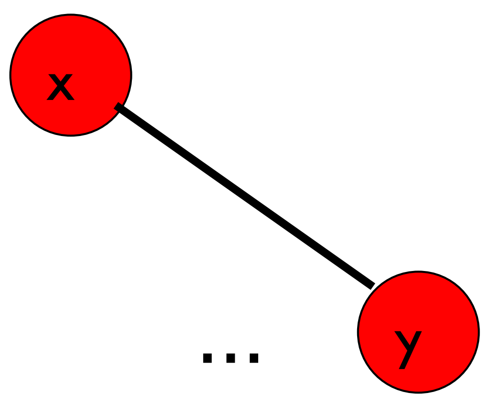
Elimination order influence

- {f(x), f(x), f(x,y)}
- Order: y, x



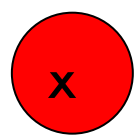
Elimination order influence

- {f(x), f(x), f(x,y)}
- Order: y, x



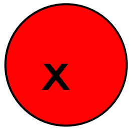
Elimination order influence

- {f(x), f(x), f(x)}
- Order: x



Elimination order influence

- $\{f(x), f(x), f(x)\}$
- Order: x

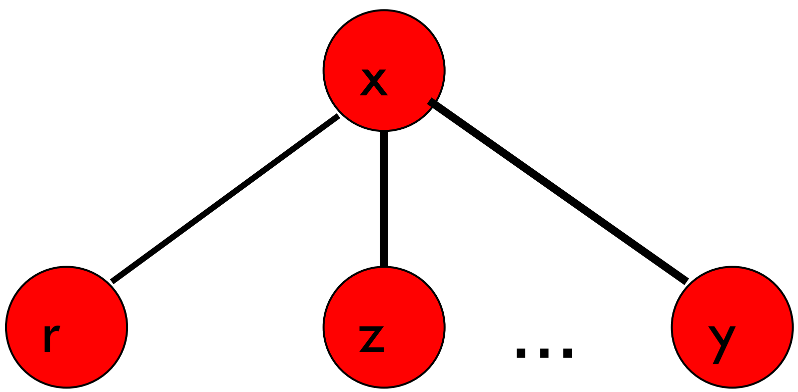


Elimination order influence

- {f()}
- Order:

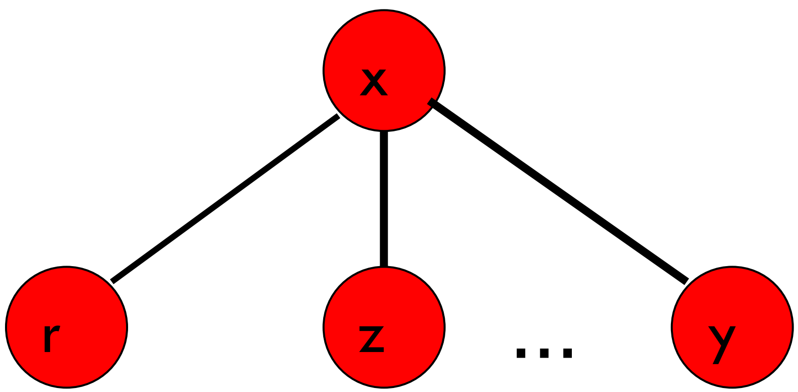
Elimination order influence

- {f(x,r), f(x,z), …, f(x,y)}
- Order: x, y, z, …, r



Elimination order influence

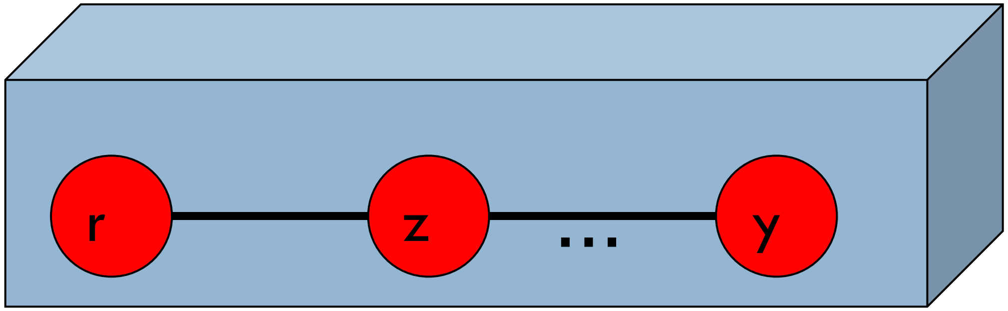
- {f(x,r), f(x,z), …, f(x,y)}
- Order: x, y, z, …, r



Elimination order influence

- {f(r,z,…,y)}
- Order: y, z, r

CLIQUE

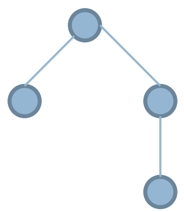


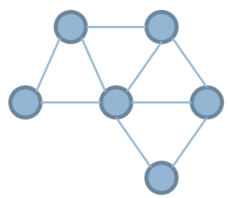
Graph induced width

- The induced width w* of a graph is its minimum induced width (over all orders)
- Existence of an order with bounded induced wwidth is NP-complete
- Trees have $w^* = 1$ (topological ordering)
- (m,n) grids have $w^* \sim \min(m,n)$ (left-right ordering)
- Linear algorithm tells if a graph has $w^* \leq k$ (fixed) (Bodlaender, 1992)

Caracterizing graphs with w* = k

- A *k*-tree is either:
  - A clique of *k* vertices
  - Obtained by adding a vertex connected to all vertices of a *k*-clique in an existing *k*-tree





- A graph with induced width *k* can be embedded in a *k*-tree (is a partial *k*-tree).

Variants of NSDP (exact computations)

- **Eliminate block by block** (Bertelé, Brioschi, 1972)
  - Same worst-case time complexity
  - Improved space complexity
  - Related to tree decompositions (Bodlaender 1994)
- **Forward-Backward/ In-out variants**
  - Computes all variable marginals in two passes
  - Cluster-tree elimination, Shenoy/Shafer, Spiegelhalter…
- **Tree-search based**
  - Recursive conditioning (Darwiche 2001)
  - AND/OR search (Dechter, Mateescu, 2007)
  - Backtrack Tree Decomposition (Jégou, Terrioux, 2003)

Historical perspective

Davis et Putnam (1960)	satisfaction
Peeling (Elston Stewart 1971)	integration
Non serial DP (Bertelé, Brioschi, 1972)	optimization
Acyclic schemes in databases (Beeri et al, 1983)	satisfaction
Directional resolution, adapt. consistency (Dechter 1987)	satisfaction
Pearl poly-tree alg. (Pearl 1988)	integration
Lauritzen/Spiegelhalter (1988)	integration
Shenoy and Shafer (1988-91)	algebraic
Bucket elimination (Dechter 1999)	general
The Generalized Distributive Law (Aji, McEliece, 2000)	algebraic
Factor graphs and… (Kschishang et al., 2001)	algebraic

Approximation & NSDP

- Mini-buckets (Dechter 1997, 2001)
  - Splits the set F_i into subsets of bounded size, each processed separately
  - Provides approximation with upper, lower bounds.

- Graph decomposition (Favier, de Givry, Jégou 2009)
  - Process each « component » and combine the results (ignoring interactions)
  - Provides bounds (#CSP)

Conclusion

- Non serial DP is a widely used approach for solving discrete optimization/integration problems
- Worst case exp time/space by induced width
- Also applies to mixed eliminations (influence diagrams, PFU networks) but elimination order is more constrained
- Can be used to provide approximate results

Bibliography

- Favier, A. de Givry, S. Jégou. Exploiting problem structure for solution counting. Proc. of CP 2009. Lisbon, Portugal.
- Dechter, R. Mini-buckets: a general scheme for generating approximations in automated reasoning. Proc. IJCAI 1997.
- Kask, K. & Dechter, R. A general scheme for automatic generation of serach heuristics from specification dependencies. Artificial Intelligence, 2001.
- Beeri, C. et al. On the desirability of acyclic database schemes. Journal of the ACM. 1983
- Lauritzen, SL. & Spiegelhalter, DJ. Local computations with probabilities on graphical strucures and their application to expert systems. JRSS B, 1988.
- Bodlaender, H. A linear time algorithm for finding tree-decompositions of small treewidth. Proc. ACM symposium on Theory of computing. 1993.
- Bodlaender, H. A tourist guide through treewidth. Develompments in Theoretical Computer Science, 1994.
- Bertelé, U. & Brioschi, F. Non Serial Dynamic Programming, Academic Press, 1972.
- Davis, Putnam, A computing procedure for quantification theory, Journal of the ACM, 1960.

Bibliography

- Darwiche, A. Recursive conditioning. Artificial Intelligence. 2001
- Dechter, A. Bucket elimination: a unifying framework for reasoning. Artificial Intelligence. 1999
- Kschischang FR. factor graphs and the sum-product algorithm. IEEE Trans. Information Theory. 2001
- Aji, SM. & McEliece, RJ. The Ggeneralized distributive law. IEEE Trans. Information Theory. 2000.
- Elston, RC. & Stewart, J. A general model for the genetic analysis of pedigree data. Human heredity. 1971
- Dechter, R. & Pearl, J. Network-based heuristics for constraint-satisfaction. Artificial Intelligence. 1987
- Shafer, GR. & Shenoy P., Local cmputations in hypertrees. Working paper 1988-91. Univ. Kansas Technical Report.
- Dechter, R., Mateescu, R. AND/OR search spaces for graphical models. Artificial Intelligence. 2007.