Exact ensemble properties in combinatorial dynamic programming schemes

Yann Ponty Cédric Saule

École Polytechnique/CNRS/INRIA AMIB – France

Formal stuffs

Optimization problem=

- Search space S
- Objective function f

Problem: Find element $e \in S$ which min(max)-imizes $f(s)$?

Dynamic programming scheme relates the minimal value of f to its minimal value(s) on some *smaller* search space(s) (Substructure property).

DP scheme = Efficient factorization (traversal) of S.
Alt.: DP scheme is generating search space.

Combinatorial dynamic programming

In order to say something general, let us be specific. . .

Definition (Combinatorial DP)

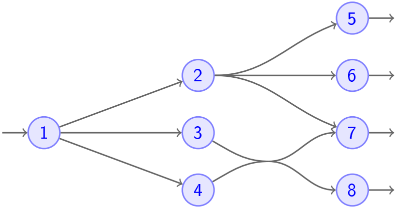
A combinatorial DP scheme computes functions that are locally additive and relies on a decomposition that is:

- Unambiguous: Each solution generated at most once!
- Complete: Each solution generated at least once!

Based on this property, DP schemes for optimization readily translate into *DP* schemes for counting, generating. . .
Remark: None of this above is strictly required by DP!

What is a decomposition?

Hypergraphs as decompositions



Hypergaphs generalize directed graphs to arcs of arbitrary in/out degrees.

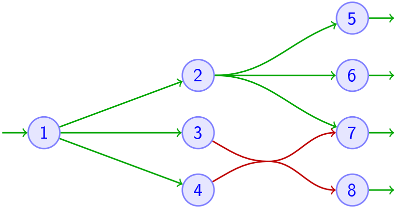
Definition (Hypergraph)

A directed hypergaph $\mathcal{H}$ is a couple (V, E) such that:

- V is a set of vertices
- E is a set of hyperarcs $e = (t(e) \to h(e))$ such that $t(e), h(e) \subset E$

Forward hypergraphs, or F-graphs, are hypergraphs whose arcs have ingoing degree exactly 1.

Hypergraphs as decompositions



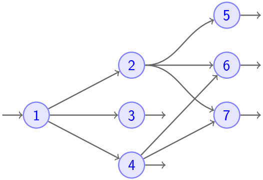
Hypergaphs generalize directed graphs to arcs of arbitrary in/out degrees.

Definition (Hypergraph)

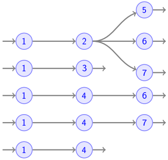
A directed hypergaph $\mathcal{H}$ is a couple (V, E) such that:

- V is a set of vertices
- E is a set of hyperarcs $e = (t(e) \to h(e))$ such that $t(e), h(e) \subset E$

Forward hypergraphs, or F-graphs, are hypergraphs whose arcs have ingoing degree exactly 1.



F-graph



All F-paths starting from vertex 1

Definition (F-path)

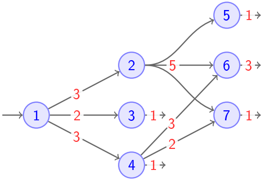
A F-path is a tree having root $s \in V$, whose children are F-paths built from the outgoing vertices of some arc $e = (s \to \mathbf{t}) \in E$.

Remark: Vertices of out degree 0 ($\mathbf{t} = \varnothing$) provide an elegant terminal case to the above recursive definition.

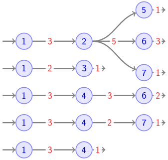
F-graph is independent iff each F-path sees at most once each arc.

A numerical valued-fonction $\pi : E \to \mathbb{R}$ can be assigned to each arc:

- Weight of a path is the product of its arcs' values
- Score of a path is the sum of its arcs' values



F-graph



All F-paths starting from vertex 1

Definition (F-path)

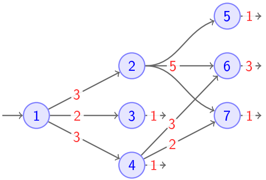
A F-path is a tree having root $s \in V$, whose children are F-paths built from the outgoing vertices of some arc $e = (s \rightarrow \mathbf{t}) \in E$.

Remark: Vertices of out degree 0 ($\mathbf{t} = \varnothing$) provide an elegant terminal case to the above recursive definition.

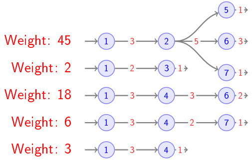
F-graph is independent iff each F-path sees at most once each arc.

A numerical valued-fonction $\pi : E \rightarrow \mathbb{R}$ can be assigned to each arc:

- Weight of a path is the product of its arcs' values
- Score of a path is the sum of its arcs' values



F-graph



All F-paths starting from vertex 1

Definition (F-path)

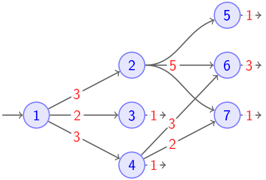
A F-path is a tree having root $s \in V$, whose children are F-paths built from the outgoing vertices of some arc $e = (s \rightarrow \mathbf{t}) \in E$.

Remark: Vertices of out degree 0 ($\mathbf{t} = \varnothing$) provide an elegant terminal case to the above recursive definition.

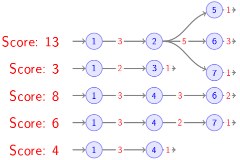
F-graph is independent iff each F-path sees at most once each arc.

A numerical valued-fonction $\pi : E \rightarrow \mathbb{R}$ can be assigned to each arc:

- Weight of a path is the product of its arcs' values
- Score of a path is the sum of its arcs' values



F-graph



All F-paths starting from vertex 1

Definition (F-path)

A F-path is a tree having root $s \in V$, whose children are F-paths built from the outgoing vertices of some arc $e = (s \to \mathbf{t}) \in E$.

Remark: Vertices of out degree 0 ($\mathbf{t} = \varnothing$) provide an elegant terminal case to the above recursive definition.

F-graph is independent iff each F-path sees at most once each arc.

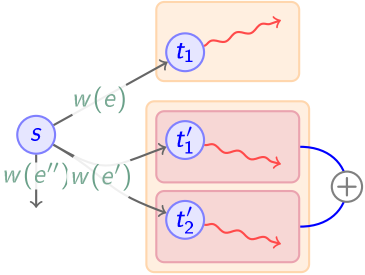
A numerical valued-fonction $\pi : E \to \mathbb{R}$ can be assigned to each arc:

- Weight of a path is the product of its arcs' values
- Score of a path is the sum of its arcs' values

Basic algorithms

$\mathcal{H} = (s_0, V, E, \pi)$: acyclic F-graph s_0: Init. node π: value fun.

$$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$$

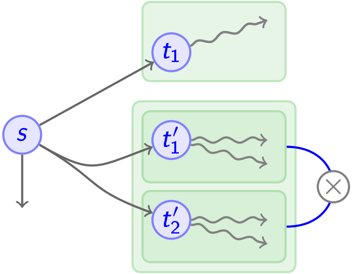


Problem	Recurrence	Complexities time/space
Min score	$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$	$\Theta(\|E\| + \|V\|)/\Theta(\|V\|)$
Num. paths	$n_s = \sum_{(s\to\mathbf{t})} \prod_{u\in\mathbf{t}} n_u$	$\Theta(\|E\| + \|V\|)/\Theta(\|V\|)$
Total weight	$w_s = \sum_{e=(s\to\mathbf{t})} w(e) \cdot \prod_{s'\in\mathbf{t}} w_{s'}$	$\Theta(\|E\| + \|V\|)/\Theta(\|V\|)$

Basic algorithms

$\mathcal{H} = (s_0, V, E, \pi)$: acyclic F-graph s_0: Init. node π: value fun.

$$n_s = \sum_{(s\to \mathbf{t})} \prod_{u\in \mathbf{t}} n_u$$

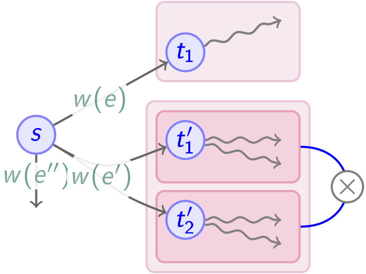


Problem	Recurrence	Complexities time/space
Min score	$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$	$\Theta(\|E\|+\|V\|)/\Theta(\|V\|)$
Num. paths	$n_s = \sum_{(s\to\mathbf{t})} \prod_{u\in\mathbf{t}} n_u$	$\Theta(\|E\|+\|V\|)/\Theta(\|V\|)$
Total weight	$w_s = \sum_{e=(s\to\mathbf{t})} w(e) \cdot \prod_{s'\in\mathbf{t}} w_{s'}$	$\Theta(\|E\|+\|V\|)/\Theta(\|V\|)$

Basic algorithms

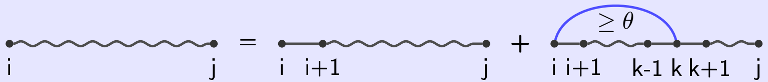
$\mathcal{H} = (s_0, V, E, \pi)$: acyclic F-graph s_0: Init. node π: value fun.

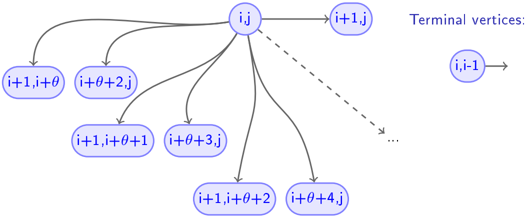
$$w_s = \sum_{e=(s \to \mathbf{t})} w(e) \cdot \prod_{s' \in \mathbf{t}} w_{s'}$$



Problem	Recurrence	Complexities time/space
Min score	$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
Num. paths	$n_s = \sum_{(s\to\mathbf{t})} \prod_{u\in\mathbf{t}} n_u$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
Total weight	$w_s = \sum_{e=(s\to\mathbf{t})} w(e) \cdot \prod_{s'\in\mathbf{t}} w_{s'}$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$

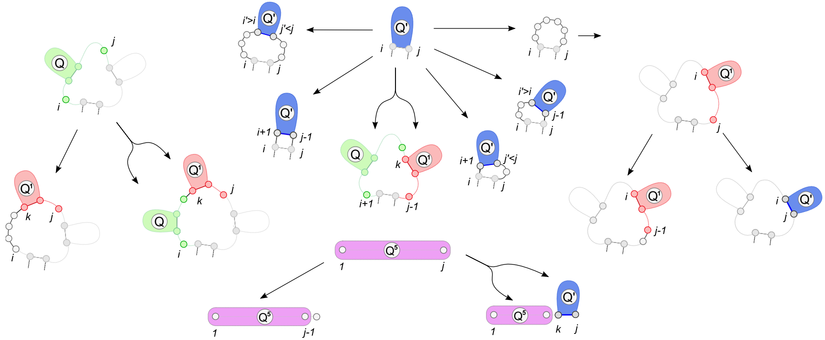
Nussinov's algorithm revisited



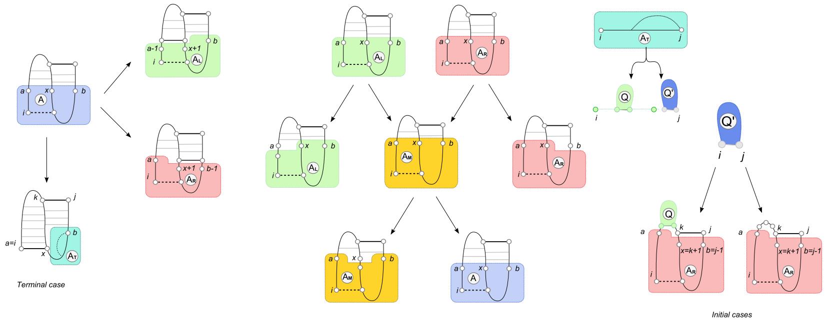


This decomposition is unambiguous!
(Proof may use length generating functions).

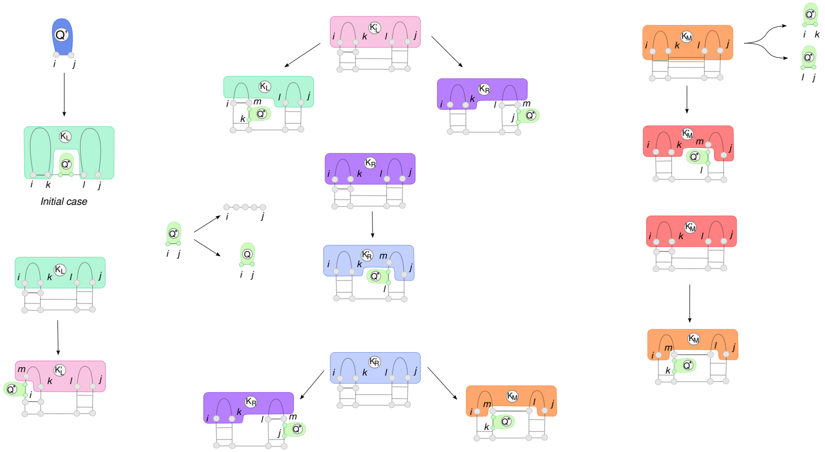
Mfold/Unafold decomposition



Akutsu's simple pseudoknots



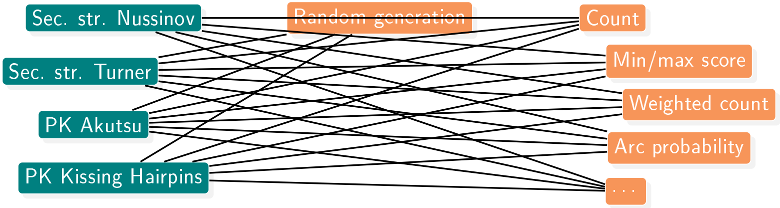
Kissing hairpins



Half time summary

Message #1

Applications of DP could (and should) be detached from the equation, and be expressed at an abstract – combinatorial – level.



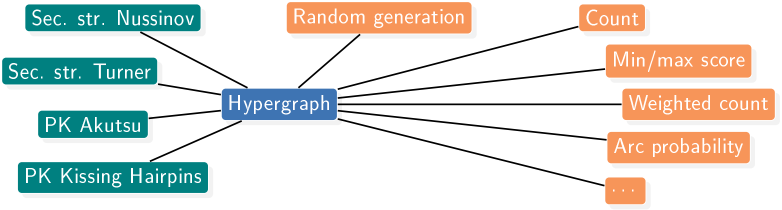
Credits: Roytberg and Finkelstein for Hypergraph DP in Bioinformatics, L. Hwang for algebraic hypergraph DP, R. Giegerich for ADP...

Let us extend applications of DP...

Half time summary

Message #1

Applications of DP could (and should) be detached from the equation, and be expressed at an abstract – combinatorial – level.



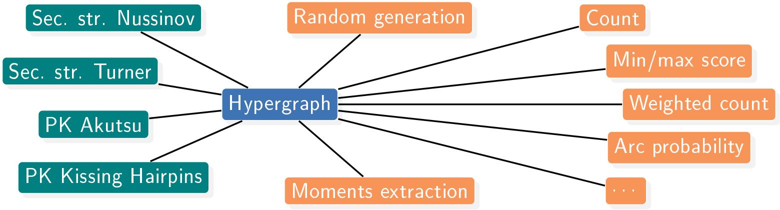
Credits: Roytberg and Finkelstein for Hypergraph DP in Bioinformatics, L. Hwang for algebraic hypergraph DP, R. Giegerich for ADP. . .

Let us extend applications of DP. . .

Half time summary

Message #1

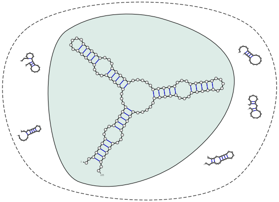
Applications of DP could (and should) be detached from the equation, and be expressed at an abstract – combinatorial – level.



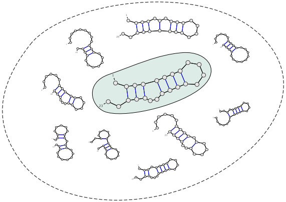
Credits: Roytberg and Finkelstein for Hypergraph DP in Bioinformatics, L. Hwang for algebraic hypergraph DP, R. Giegerich for ADP...

Let us extend applications of DP...

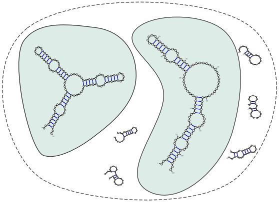
Distribution of solutions



Functional folding?



Ill-defined folding: mRNA?



Bistable RNA Kinetics?

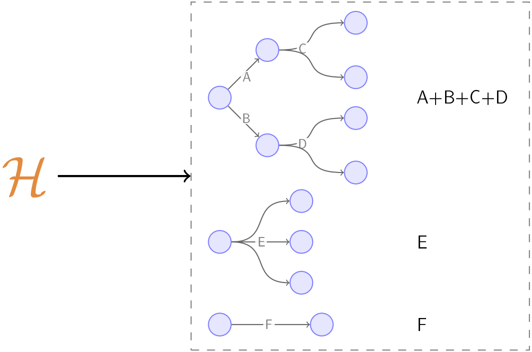
- An acyclic F-graph $\mathcal{H} = (s_0, V, E, \pi)$, defining a *search space* $\mathcal{T}$ the set of F-paths (trees) in $\mathcal{H}$.
- Feature functions $\alpha_1, \ldots, \alpha_k : E \to \mathbb{R}$ extended **additively** on $\mathcal{T}$.

What can we say about the distribution(s) of α_i(s)?

Naive approach: Compute the distribution exactly, accessible values being choices of $\mathcal{O}(n)$ values among $|E|$, n =#arcs in largest F-path.
$\Rightarrow$ DP count the #ways of accessing each value in exponential time...

Example: Mean value of a feature function

What is the average values of α assuming a uniform distribution on $\mathcal{T}$?

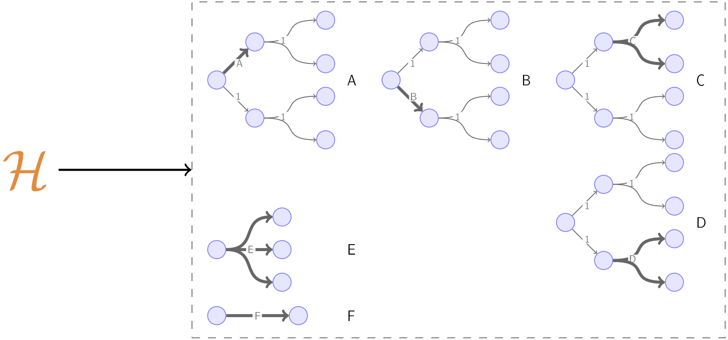


$$\text{Mean value} = \text{Weighted sum} = \frac{\sum_{t\in\mathcal{T}} \alpha(t)}{|\mathcal{T}|}$$

Remark: Dropping the terminal edges (Alt. $\alpha(e) = 0$)...

Example: Mean value of a feature function

What is the average values of α assuming a uniform distribution on $\mathcal{T}$?

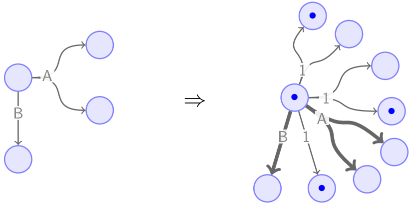


$$\text{Mean value} = \text{Weighted sum} = \frac{\sum_{t\in\mathcal{T}}\alpha(t)}{|\mathcal{T}|}$$

Principle and main equation

By introducing a controlled ambiguity, one can extract the feature expectation from a combinatorial DP scheme.

Idea: Passing a dot • along or drop it ($\equiv \alpha(e)$) on some arc e.



$$\Rightarrow c_s^{\bullet\alpha} = \sum_{o\in\mathcal{T}} \alpha(o) = \sum_{e=(s\to\mathbf{t})} \left(\alpha(e)\cdot\prod_{t_i\in\mathbf{t}} c_{t_i} + \sum_{t_i\in\mathbf{t}} c_{t_i}^{\bullet\alpha} \prod_{t_j\neq t_i\in\mathbf{t}} c_{t_j} \right)$$

$\Rightarrow \mathbb{E}(\alpha) := c_{s_0}^{\bullet\alpha}/c_{s_0}$ obtained in $\Theta(|V| + |E|\cdot D^2)/\Theta(|V|)$ time/space, with $D := \max_{e\in E}(|h(e)|)$

Remark: Similar to the *folklore* pointing operator in enumerative combinatorics, and to the formal derivative.

Low-hanging fruits

D: $\max_{e\in E}(|h(e)|)$

- **Boltzmann-like distribution**
  Unconvinced by the uniform distribution? Got an energy fonction $\Delta : E \to \mathbb{R}$ extended additively, and a good reason to expect convergence toward a Boltzmann distribution $e^{\frac{-\Delta}{kT}}$? (We do...)
  $\Rightarrow$ Expectation in the Boltzmann distribution is just a weight away.

$$c_s^{\bullet\alpha} = \sum_{o\in\mathcal{T}} \alpha(o)\cdot e^{-\Delta(o)/RT}$$

$$= \sum_{e=(s\to\mathbf{t})} \mathbf{e}^{\frac{-\Delta(\mathbf{e})}{\mathbf{kT}}} \left(\alpha(e)\cdot\prod_{t_i\in\mathbf{t}} c_{t_i} + \sum_{t_i\in\mathbf{t}} c_{t_i}^{\bullet\alpha} \prod_{t_j\neq t_i\in\mathbf{t}} c_{t_j}\right)$$

$\Rightarrow \Theta(|V| + |E|\cdot D^2)/\Theta(|V|)$

- Higher-order moments
- Exact correlations

Nb: Specializes into Miklos *et al* 2005 for RNA secondary structures.

Low-hanging fruits

D: $\max_{e\in E}(|h(e)|)$

- Boltzmann-like distribution $\Rightarrow \Theta(|V| + |E| \cdot D^2)/\Theta(|V|)$
- Higher-order moments
  Want to go beyond the expected value of α? No problem (almost), compute higher-order moments.

$$c_s^{\bullet^m_\alpha} = \sum_{o\in\mathcal{T}} \alpha(o)^m \cdot e^{-\Delta(o)/RT}$$

$$= \sum_{e=(s\to\mathbf{t})} e^{\frac{-\Delta(e)}{kT}} \left(\sum_{\substack{(m'_1,\cdots,m'_{|\mathbf{t}|})\\ \text{s.t.} \sum m'_i := m' \leq m}} \alpha(e)^{m-m'} \prod_{t_i\in\mathbf{t}} c_{t_i}^{\bullet^{m'_i}_\alpha} \right)$$

$\Rightarrow \Theta(|V| + |E| \cdot m^{D+2})/\Theta(m \cdot |V|)$

- Exact correlations

Nb: Specializes into Miklos *et al* 2005 for RNA secondary structures.

Low-hanging fruits

D: $\max_{e\in E}(|h(e)|)$

- Boltzmann-like distribution $\Rightarrow \Theta(|V| + |E| \cdot D^2)/\Theta(|V|)$
- Higher-order moments $\Rightarrow \Theta(|V| + |E| \cdot m^{D+2})/\Theta(m \cdot |V|)$
- Exact correlations Given two α and α', correlation is given by

$$c_s^{\bullet_\alpha \bullet_{\alpha'}} = \sum_{o\in\mathcal{T}} \alpha(o)\cdot\alpha'(o)\cdot e^{-\Delta(o)/RT}$$

$$= \sum_{e=(s\to\mathbf{t})} e^{\frac{-\Delta(e)}{kT}} \left(\sum_{\substack{((m_1\cdot m_{|\mathbf{t}|}),\\ (m'_1\cdot m'_{|\mathbf{t}|}))\\ m:=\sum m_i\leq 1,\\ m':=\sum m'_i\leq 1}} \alpha(e)^{1-m}\alpha'(e)^{1-m'} \prod_{t_i\in\mathbf{t}} c_{t_i}^{\bullet_\alpha^{m_i}\bullet_{\alpha'}^{m'_i}} \right)$$

$\Rightarrow \Theta(|V| + |E| \cdot D^2)/\Theta(|V|)$

Nb: Specializes into Miklos *et al* 2005 for RNA secondary structures.

Conclusion

- Many algorithms in Bioinformatics rely on dynamic programming
- Adopting a combinatorial vision over the search space greatly, and hypergraph representations, helped:
  - Prove correctness.
  - Instant transposal of new applications.
- Implementation tricky (avoid explicit representations).
- Use in combination with machine-learning as classifier/scanner for ncRNAs.
- How to transpose usual DP tricks (four-russians, cutting corners, sparsification)?
- Are there even more compact ways to describe more specific search spaces (e.g. CFGs, split-types??)