Probabilistic XML: Survey and Challenges

Pierre Senellart

INRIA Lille – Nord Europe

20 November 2009

Outline

Uncertain data

Numerous sources of uncertain data:

- Measurement errors
- Data integration from contradicting sources
- Imprecise mappings between heterogeneous schemata
- Imprecise automatic process (information extraction, natural language processing, etc.)
- Imperfect human judgment

Managing this imprecision

Objective

Not to pretend this imprecision does not exist, and manage it as rigorously as possible throughout a long, automatic and human, potentially complex, process.

Especially:

- Use probabilities to represent the confidence in the data
- Query data and retrieve probabilistic results
- Allow adding, deleting, modifying data in a probabilistic way
- (If possible) Keep throughout the process lineage/provenance information, so as to ensure traceability

Managing this imprecision

Objective

Not to pretend this imprecision does not exist, and manage it as rigorously as possible throughout a long, automatic and human, potentially complex, process.

Especially:

- Use probabilities to represent the confidence in the data
- Query data and retrieve probabilistic results
- Allow adding, deleting, modifying data in a probabilistic way
- (If possible) Keep throughout the process lineage/provenance information, so as to ensure traceability

Why XML?

- Extensive literature about probabilistic relational databases [?, ?, ?]
- Different typical querying languages: conjunctive queries vs tree-pattern queries (possibly with joins)
- Cases where a tree-like model might be appropriate:
  - No schema or few constraints on the schema
  - Independent modules annotating freely a content warehouse
  - Inherently tree-like data (e.g., mailing lists, parse trees) with naturally occurring queries involving the descendant axis

Remark

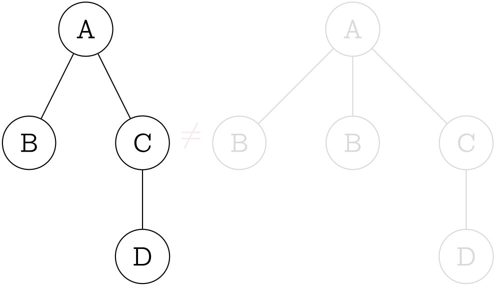
Some results can be transferred from one model to the other. In other cases, connection much trickier (see later)!

Outline

Outline

Trees and possible worlds

Unordered **data trees**



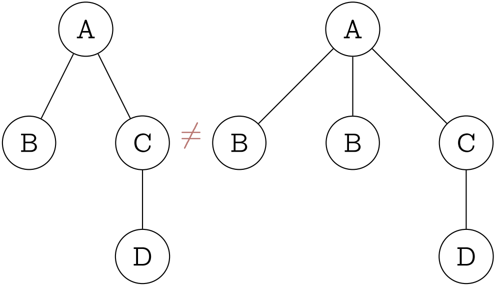
(multiset semantics)

Sample space: Set of all such data trees.

Probabilistic XML database: (Succinct) representation of a discrete probability distribution over this sample space (= a set of possible worlds).

Trees and possible worlds

Unordered data trees



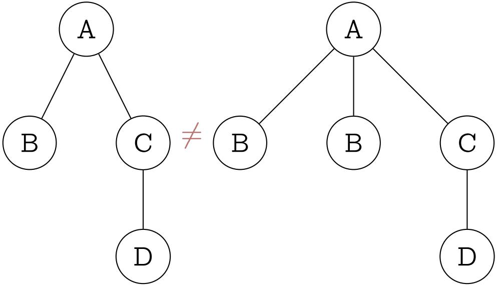
(multiset semantics)

Sample space: Set of all such data trees.

Probabilistic XML database: (Succinct) representation of a discrete probability distribution over this sample space (= a set of possible worlds).

Trees and possible worlds

Unordered data trees

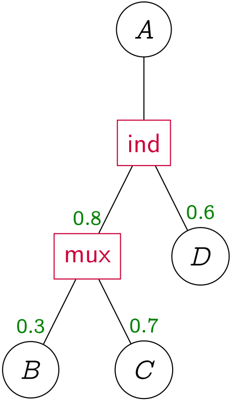


(multiset semantics)

Sample space: Set of all such data trees.

Probabilistic XML database: (Succinct) representation of a discrete probability distribution over this sample space (= a set of possible worlds).

Local dependencies [?, ?]



- Tree with ordinary (circles) and distributional (rectangles) nodes
- Distributional nodes specify how their children can be randomly selected (here, independently or in a mutually exclusive way)
- Possible-world semantics: every possible selection of children of distributional nodes, with associated probability
- No long-distance probabilistic dependencies in the tree!

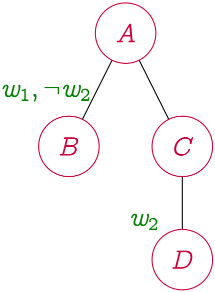
Types of distributional nodes [?, ?]

- **det** all children of the node are deterministically selected
- **ind** children of the node are chosen independently of one another, according to their probabilities
- **mux** children of the node are chosen in a mutually exclusive way, depending of their probabilities, that must sum up to 1 or less
- **exp** the distribution of all possible choices of children is explicitly given: each subset of the set of the children is associated with a probability, these probabilities summing up to 1

Remark

Clearly, det is a particular case of ind, and mux is a particular case of exp.

Arbitrary dependencies: event conjunctions [?]

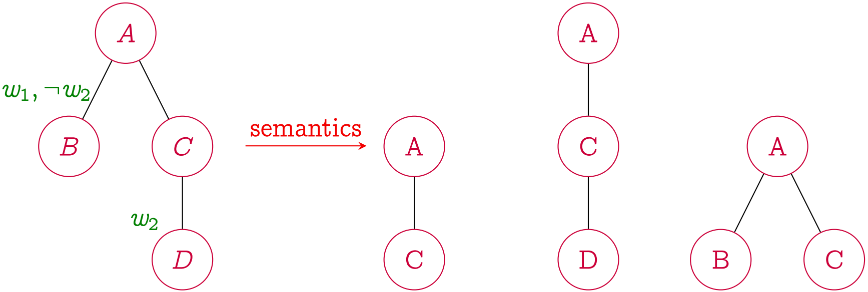


Event	Prob.
w_1	0.8
w_2	0.7

$p_1 = 0.06$ $p_2 = 0.70$ $p_3 = 0.24$

- Conjunctions of independent events on each node of the tree [?]
- Expresses arbitrarily complex dependencies
- Both ind and mux can be seen as particular cases (but not exp!)

Arbitrary dependencies: event conjunctions [?]



Event	Prob.
w_1	0.8
w_2	0.7

$p_1 = 0.06$ $p_2 = 0.70$ $p_3 = 0.24$

- Conjunctions of independent events on each node of the tree [?]
- Expresses arbitrarily complex dependencies
- Both ind and mux can be seen as particular cases (but not exp!)

Events and lineage

- Event variables: can represent the provenance of data
- Typically:
  1. At each (probabilistic) update, a new event variable is introduced
  2. Query results are given with probabilities, but also with the lineage of the query [?]
- Allow to keep track, with no additional cost, of the provenance of data!

Previously studied XML models

ProTDB [?] ind + mux

Probabilistic XML [?] mux + det, with alternation between the two kinds of nodes

SP trees [?], PEPX [?] ind without hierarchies of distributional nodes

PXML [?] exp without hierarchies, extended to graphs

Probabilistic interval XML [?] exp without hierarchies, when intervals are collapsed into points

Prob-trees [?, ?] Event conjunctions

Expressiveness

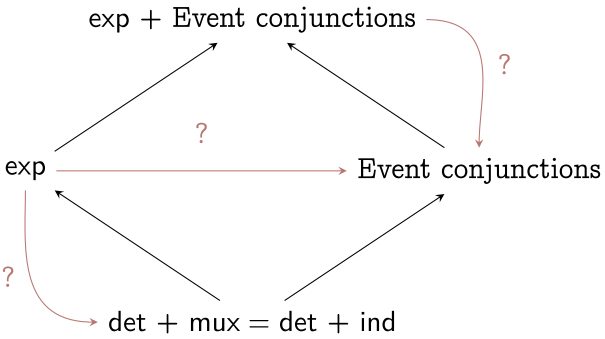
Theorem ([?, ?, ?])

1. ind *alone, or* mux *alone, are not a complete representation system.*
2. det + mux *is enough to have full expressive power.* *Consequently,* ind + mux, exp *alone, or event conjunctions, have full expressive power.*
3. *Hierarchies (allowing a distributional node below another distributional node) are important.*

Tractable reductions between models [?, ?]

$$\begin{tikzcd}
 & \text{exp} + \text{Event conjunctions} & \\
\text{exp} \arrow[ur] & & \text{Event conjunctions} \arrow[ul] \\
 & \text{det} + \text{mux} = \text{det} + \text{ind} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Tractable reductions between models [?, ?]



Outline

Querying probabilistic XML

Semantics of a (Boolean) query = probability:

1. Generate all possible worlds of a given probabilistic document
2. In each world, evaluate the query
3. Add up the probabilities of the worlds that make the query true

EXPTIME algorithm! Can we do better, i.e., can we apply directly the algorithm on the probabilistic document?

We shall talk about data complexity of query answering.

Querying probabilistic XML

Semantics of a (Boolean) query = probability:

1. Generate all possible worlds of a given probabilistic document (possibly exponentially many)
2. In each world, evaluate the query
3. Add up the probabilities of the worlds that make the query true

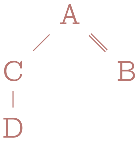
EXPTIME algorithm! Can we do better, i.e., can we apply directly the algorithm on the probabilistic document?

We shall talk about data complexity of query answering.

Boolean query languages on trees

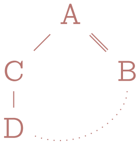
Single-path queries (SP) `/A//B/C` (no branching)

Tree-pattern queries (TP) `/A[C/D]//B`



Tree-pattern queries with joins (TPJ) `for $x in $doc/A/C/D`

`return $doc/A//B[.=$x]`



Monadic second-order queries (MSO) generalization of TP, do not cover TPJ unless the size of the alphabet is bounded

The #P and $FP^{\#P}$ complexity classes

- A (counting) problem is in #P if there is a PTIME non-deterministic Turing machine whose number of accepting paths, given as input the input of the problem, is the output of the problem.
- A problem is #P-hard if any #P problem can be PTIME-reduced to it (via a Karp reduction). #2DNF, the problem of counting the number of assignments satisfying a formula in 2-DNF, is #P-complete.
- A (computation) problem is in $FP^{\#P}$ if it is computable by a PTIME Turing machine with access to a $\#P$ oracle.
- A problem is $FP^{\#P}$-hard if any $FP^{\#P}$ problem can be PTIME-reduced to it (via a Turing reduction). Equivalently, a computation problem is $FP^{\#P}$-hard if it is #P-hard.

The #P and $\mathrm{FP}^{\#\mathrm{P}}$ complexity classes

- A (counting) problem is in #P if there is a PTIME non-deterministic Turing machine whose number of accepting paths, given as input the input of the problem, is the output of the problem.
- A problem is #P-hard if any #P problem can be PTIME-reduced to it (via a Karp reduction). #2DNF, the problem of counting the number of assignments satisfying a formula in 2-DNF, is #P-complete.

- A (computation) problem is in $\mathrm{FP}^{\#\mathrm{P}}$ if it is computable by a PTIME Turing machine with access to a $\#P$ oracle.
- A problem is $\mathrm{FP}^{\#\mathrm{P}}$-hard if any $\mathrm{FP}^{\#\mathrm{P}}$ problem can be PTIME-reduced to it (via a Turing reduction). Equivalently, a computation problem is $\mathrm{FP}^{\#\mathrm{P}}$-hard if it is #P-hard.

Complexity of query evaluation

	Local dependencies	Arbitrary dependencies
SP	PTIME	$FP^{\#P}$-complete [?]
TP	PTIME [?, ?, ?]	$FP^{\#P}$-complete
TPJ	$FP^{\#P}$-complete	$FP^{\#P}$-complete
MSO	PTIME [?]	$FP^{\#P}$-complete

Remark

Project-free queries are tractable with arbitrary dependencies. [?]

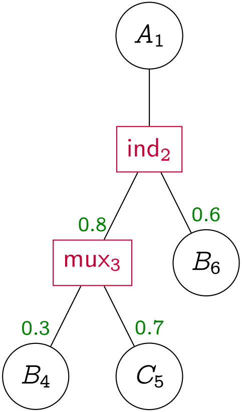
Complexity of query evaluation

	Local dependencies	Arbitrary dependencies
TP	PTIME [?, ?, ?]	
TPJ	$FP^{\#P}$-complete	

TP PTIME for local dependencies [?]

Bottom-up dynamic programming algorithm.

Query: /A//B



	A_1	ind_2	mux_3	B_4	C_5	B_6
/B				1	0	1
//B				1	0	1
/A//B				0	0	0

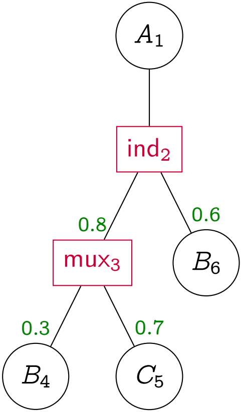
mux convex sum

ind inclusion-exclusion

TP PTIME for local dependencies [?]

Bottom-up dynamic programming algorithm.

Query: /A//B



	A_1	ind_2	mux_3	B_4	C_5	B_6
/B			0.3	1	0	1
//B			0.3	1	0	1
/A//B			0	0	0	0

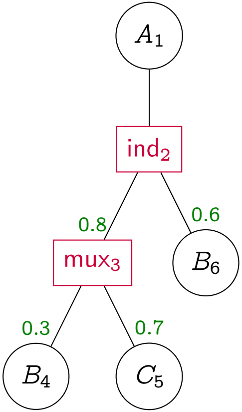
mux convex sum

ind inclusion-exclusion

TP PTIME for local dependencies [?]

Bottom-up dynamic programming algorithm.

Query: /A//B



	A_1	ind$_2$	mux$_3$	B_4	C_5	B_6
/B		0.696	0.3	1	0	1
//B		0.696	0.3	1	0	1
/A//B	0	0	0	0	0	

mux convex sum

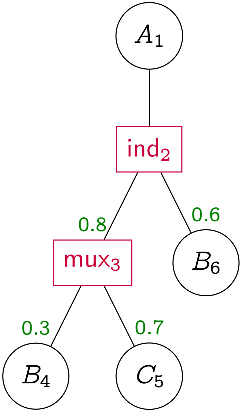
ind inclusion-exclusion

$$\begin{aligned}\Pr(\mathsf{ind}_2 \models /B) &= 1-(1-0.8\times\Pr(\mathsf{mux}_3 \models /B))\times(1-0.6\times\Pr(B_6 \models /B))\\ &= 1-(1-0.8\times 0.3)\times(1-0.6) = 0.696\end{aligned}$$

TP PTIME for local dependencies [?]

Bottom-up dynamic programming algorithm.

Query: /A//B



	A_1	ind$_2$	mux$_3$	B_4	C_5	B_6
/B	0	0.696	0.3	1	0	1
//B	0.696	0.696	0.3	1	0	1
/A//B	0.696	0	0	0	0	0

mux convex sum

ind inclusion-exclusion

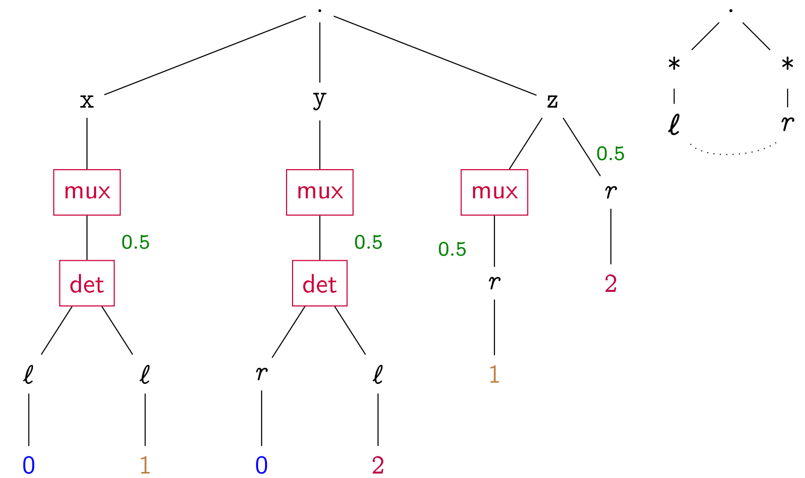
Complications [?, ?, ?]

General case:

- Branching patterns: need to consider all conjunctions of subbranches of a pattern (exponentially many!)
- Works also with exp, but more complicated inclusion-exclusion
- Number of optimizations possible
- Bottomline: it works because bottom-up evaluation is possible
- Generalization [?]: MSO queries can be converted (non efficiently) into bottom-up tree automota, therefore MSO is also tractable

TPJ $\mathrm{FP}^{\#\mathrm{P}}$-hard for local dependencies [?]

Reduction from #2DNF. Example: $\varphi = xy \lor x\neg z \lor yz$.



Aggregate Queries [?, ?]

Aggregate Queries: sum, count, avg, countd, min, max, etc.
Distributions? Possible values? Expected value?

Summary of results

- Computing expected values of sum and count tractable with arbitrary dependencies. Everything else intractable.
- Computing expected values of every of these aggregate functions is tractable with local dependencies.
- Computing distributions and possible values is tractable for count, min, max, intractable for the others.

Always possible to approximate query answers with Monte Carlo sampling.

Outline

Typing [?]

- Determining the probability that a probabilistic document with local dependencies matches a schema is tractable (uses the transformation of schemas into bottom-up automata).
- Determining the probability that a probabilistic document with arbitrary dependencies matches a schema is intractable.

Updates [?, ?]

Updates defined by a query (cf. XUpdate, XQuery Update).
Semantics: for all matches of a query, insert or delete a node in the tree at a place located by the query.

Results

- Updates are intractable with local dependencies: the result of an update can require an exponentially larger representation size
- Insertions are tractable with arbitrary dependencies; deletions are intractable.

Outline

Missing complexity results

- Tractable reduction from exp to arbitrary dependencies?
- Tractable reduction from exp to mux + ind?
- More systematic study of updates (different semantics: insert once if there is at least one match).
- Combined complexity results.

Work in progress with U. Oxford and U. Bozen-Bolzano

Link with probabilistic relational models

Relational case
(Block-independent disjoint model, [?])

- Some conjunctive queries are PTIME
- Others are #P-hard
- Complex conditions to separate the two

XML case (Local dependencies)

- Tree pattern queries are PTIME
- Tree pattern queries with (non-trivial) joins are #P-hard

- Why does the XML case seem simpler?
- Is there some insight to be gained from one case to the other?
- Translating XML data and queries to the relational case yields queries with self-joins, a less well-understood setting

Link with probabilistic relational models

Relational case (Block-independent disjoint model, [?])

- Some conjunctive queries are PTIME
- Others are #P-hard
- Complex conditions to separate the two

XML case (Local dependencies)

- Tree pattern queries are PTIME
- Tree pattern queries with (non-trivial) joins are #P-hard

- Why does the XML case seem simpler?
- Is there some insight to be gained from one case to the other?
- Translating XML data and queries to the relational case yields queries with self-joins, a less well-understood setting

Continuous probability distributions

- Most probabilistic database models assume discrete probabilistic distributions
- Sensor networks, unknown values: need for continuous distributions! (uniform, Gaussian, Poisson, etc.)
- Some existing works on query answering over continuous distributions [?, ?] but no clear semantics
- Claim: this is not more difficult than the discrete case, as long as integration/differentiation are easy (symbolically or numerically) for the considered distributions
- Discrete distributions can be modeled as Diracs

Work in progress with U. Bozen-Bolzano [?]

Tractable extensions of the local dependency model

- Arbitrary dependencies: not tractable
- Local dependencies: not practical
- Somewhere in between?
  - What makes the arbitrary dependency model hard?
  - How can the local dependency model be generalized, while remaining tractable?
- And can we go further? cf. XML schemas
  - Trees of unbounded depth
  - Trees of unbounded width
  - Infinite trees?

Work in progress with U. Oxford

But where do probabilities come from?!

- Do the numbers assigned as probabilities in PDBMS really make sense?
- In some cases, sources of “good” probabilities:
  - Statistics
  - Conditional Random Fields
- What about the rest? Does it really make sense to model uncertainty with probabilities?

A system that just works

- Nothing else than toy systems exist for probabilistic XML
- What should it be based upon:
  - a probabilistic relational DBMS?
  - a native XML DBMS?
- Systems issue: distribution, indexing, etc.
- And need for a killer application!
  - Probabilistic content warehouse?
  - Parse trees of natural language sentences?
  - Concise representation of a large corpus of XML documents?

PhD started on this topic in October 2009

Merci.

Foundations of Web data management

`http://webdam.inria.fr/`

References I

Serge Abiteboul, T-H. Hubert Chan, Evgeny Kharlamov, Werner Nutt, and Pierre Senellart.
Aggregate queries for discrete and continuous probabilistic xml.
In *Proc. ICDT*, Lausanne, Switzerland, March 2010.

Serge Abiteboul, Benny Kimelfeld, Yehoshua Sagiv, and Pierre Senellart.
On the expressiveness of probabilistic XML models.
VLDB Journal, 18(5):1041–1064, October 2009.

Serge Abiteboul and Pierre Senellart.
Querying and updating probabilistic information in XML.
In *Proc. EDBT*, Munich, Germany, March 2006.

References II

Reynold Cheng, Dmitri V. Kalashnikov, and Sunil Prabhakar.
Evaluating probabilistic queries over imprecise data.
In *Proc. SIGMOD*, San Diego, CA, USA, June 2003.

Sara Cohen, Benny Kimelfeld, and Yehoshua Sagiv.
Incorporating constraints in probabilstic XML.
In *Proc. PODS*, Vancouver, BC, Canada, June 2008.

Sara Cohen, Benny Kimelfeld, and Yehoshua Sagiv.
Running tree automata on probabilistic XML.
In *Proc. PODS*, Providence, RI, USA, June 2009.

Amol Deshpande, Carlos Guestrin, Samuel Madden, Joseph M. Hellerstein, and Wei Hong.
Model-driven data acquisition in sensor networks.
In *Proc. VLDB*, Toronto, ON, Canada, August 2004.

References III

- Nilesh Dalvi, Chrisopher Ré, and Dan Suciu.
  Probabilistic databases: Diamonds in the dirt.
  Communications of the ACM, 52(7), 2009.
- Nilesh N. Dalvi and Dan Suciu.
  Management of probabilistic data: foundations and challenges.
  In *Proc. PODS*, Beijing, China, June 2007.
- J. Nathan Foster, Todd J. Green, and Val Tannen.
  Annotated XML: queries and provenance.
  In *Proc. PODS*, Vancouver, BC, Canada, June 2008.
- Edward Hung, Lise Getoor, and V. S. Subrahmanian.
  PXML: A probabilistic semistructured data model and algebra.
  In *Proc. ICDE*, Bangalore, India, March 2003.

References IV

Edward Hung, Lise Getoor, and V. S. Subrahmanian.
Probabilistic interval XML.
TOCL, 8(4), 2007.

Tomasz Imieliński and Witold Lipski.
Incomplete information in relational databases.
Journal of the ACM, 31(4):761–791, 1984.

Benny Kimelfeld, Yuri Kosharovsky, and Yehoshua Sagiv.
Query efficiency in probabilistic XML models.
In *Proc. SIGMOD*, Vancouver, BC, Canada, June 2008.

Benny Kimelfeld, Yuri Kosharovsky, and Yehoshua Sagiv.
Query evaluation over probabilistic XML.
VLDB Journal, 18(5):1117–1140, October 2009.

References V

- Christoph Koch.
  MayBMS: A system for managing large uncertain and probabilistic databases.
  In Charu Aggarwal, editor, *Managing and Mining Uncertain Data*. Springer-Verlag, 2009.

- B. Kimelfeld and Y. Sagiv.
  Matching twigs in probabilistic XML.
  In *Proc. VLDB*, Vienna, Austria, September 2007.

- Te Li, Qihong Shao, and Yi Chen.
  PEPX: a query-friendly probabilistic XML database.
  In *Proc. CIKM*, Arlington, VA, USA, November 2006.

References VI

Andrew Nierman and H. V. Jagadish.
ProTDB: Probabilistic data in XML.
In *Proc. VLDB*, Hong Kong, China, August 2002.

Pierre Senellart and Serge Abiteboul.
On the complexity of managing probabilistic XML data.
In *Proc. PODS*, Beijing, China, June 2007.

Maurice van Keulen, Ander de Keijzer, and Wouter Alink.
A probabilistic XML approach to data integration.
In *Proc. ICDE*, Tokyo, Japan, April 2005.

Jennifer Widom.
Trio: A system for integrated management of data, accuracy, and lineage.
In *Proc. CIDR*, Asilomar, CA, USA, January 2005.