Stat 232B- CS266B

Statistical Computing and Inference in Vision and Image Science

Lecture 1 Introduction

1, Background: 3 model regimes x 2 objectives

In computer science, we know that
representations or models are often of prime importance,
while algorithms are designed for certain representations. Therefore,
before we study algorithms, we need to know what are the typical representations.

In Stat232A, we represent all models by probabilities defined on graphical representations, and divide them into three regimes:

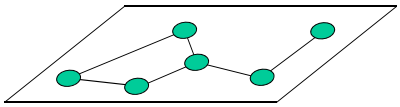
Regime 1: plat graph where all vertices are semantically at the same level,
e.g. Markov random fields, Gibbs models for systems of large degrees of freedom.

Regime 2: hierarchical graph where a high level node is divided into various components at the low level, e.g. Markov trees, sparse coding, stochastic context free grammars.

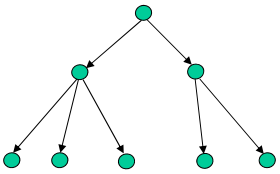
Regime 3: integrating models in regime 1 and 2 in general and-or graph representation.

Three regimes of models from Stat232A

Regime 1, Flat: Descriptive or declarative
(Constraint-satisfaction, Markov random fields, Gibbs, Julesz ensemble, Contextual)



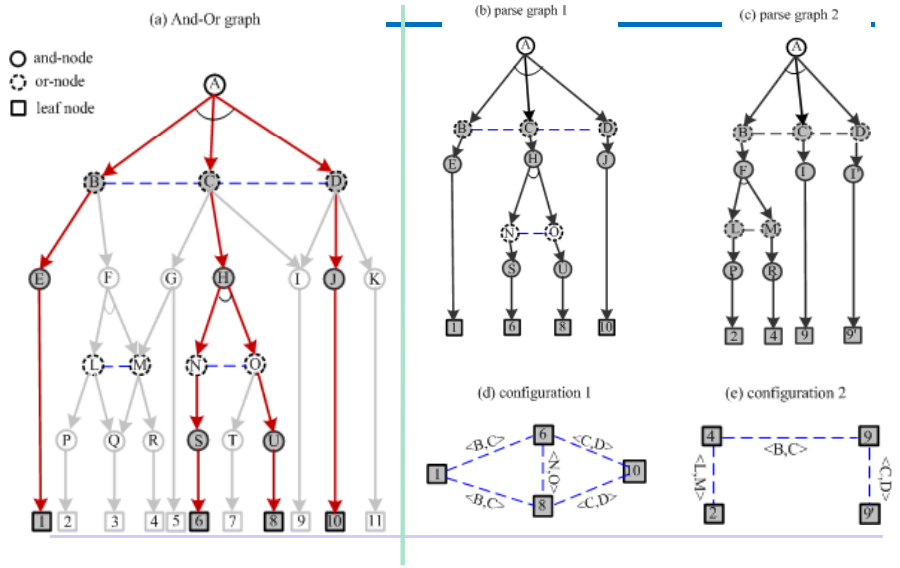
Regime 2, Hierarchical: Generative or compositional
(Markov tree, stochastic context free grammar, sparse coding)



Regime 3, Integrated: hierarchy + context
(And-or graphs, stochastic Context Sensitive Grammar).

See next page for an example.

And-Or graph, parse graphs, and configurations



Objective of algorithms design

Generally speaking, there are two types of objectives in the literature.

Objective 1: *seeking joint optimal solution for all nodes in the graph simultaneously*
such as image segmentation, scene labeling --- on flat graphs;
image parsing, event parsing --- on hierarchical graphs.

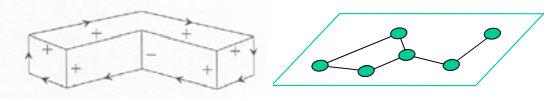
Objective 2: *seeking marginal optimal solution for certain node in the graph*
such as classification or detection of objects.
Although the algorithms use all features on and off the object, they do not explicitly solve for the other variables, such as parts etc. The nodes interact through local non-maximum suppression.

In the past decade, the vision and learning community focused on the marginal tasks.

In this class, we will also study a computing paradigm that uses the marginal information to drive the joint optimization tasks ---e.g. Data Driven Markov chain Monte Carlo.

Algorithms: A tale of three kingdoms

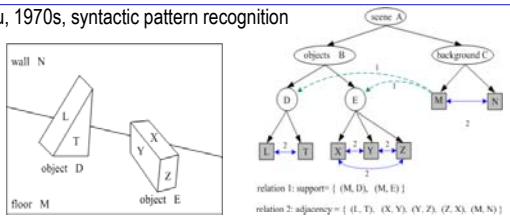
Waltz, 1960s constrain - satisfaction



Flat descriptive models
Markov random fields
Graphical models
FRAME, Mixed Random fields

--- contexts at all levels

Fu, 1970s, syntactic pattern recognition

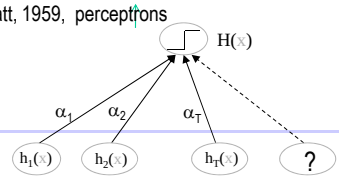


Hierarchic generative models

Stochastic context free grammar
Sparse coding
Wavelets / harmonic analysis
image grammars

--- vocabulary at all levels

Rosenblatt, 1959, perceptrons



Discriminative models

Adaboosting

--- features at all levels

Algorithms for the three kingdoms

We organize these algorithms in three methods according to the underlying representation.

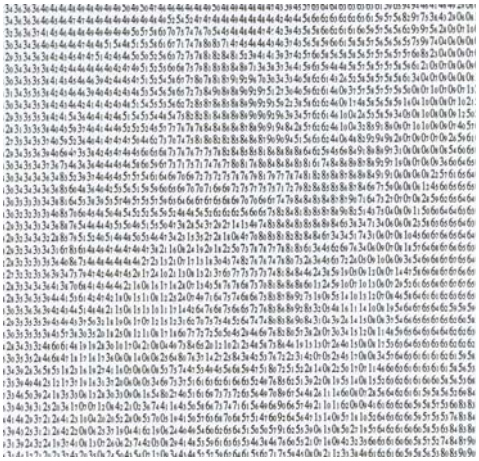
1. *Descriptive methods*: algorithms working on various graphs where the nodes/vertices represent states of the same semantic level. E.g, constraint-satisfaction, relaxation-labeling, dynamic programming, Viterbi, belief propagation, Gibbs sampler, Sequential Monte Carlo, Swendsen-Wang cuts.

2. *Generative methods:* algorithms working on hierarchic graph representations where one level of vertices semantically generate the nodes at the level below as parts/components. E.g. heuristic search, search on And-Or graphs, matching pursuit, various parsing algorithms (inside-outside, Earley), Metropolis-Hastings, Markov chain Monte Carlo, reversible jumps.

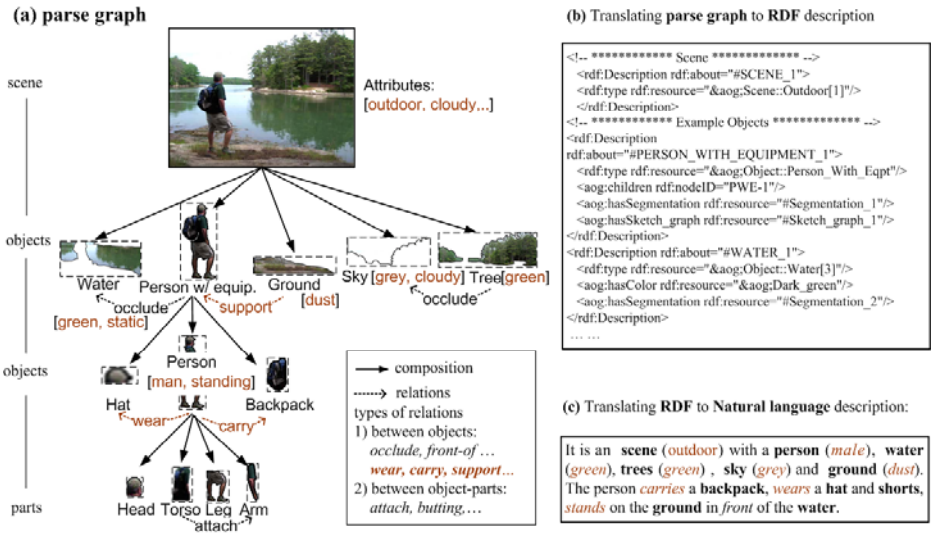
3. *Discriminative methods*: algorithm working on selecting features for discriminating various classes. taught in stat231, e.g. clustering, decision tree, boosting, SVM.

Vision: understanding the contents of images

The task of understanding images is also called *image parsing* and is posed as a joint inference problem, involving hierarchical and contextual representations.



An example of image parsing and text description



Yao et al. I2T: From Image parsing to text generation, Proceedings of IEEE, 2010.

Joint Spatial, Temporal, Causal and Text Parsing

See demo and query at

http://vcla.stat.ucla.edu/see/demo.html

Supported by ONR MURI and DARPA MSEE

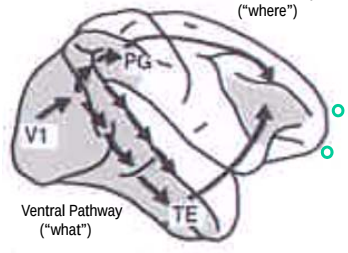
2, Observations of the vision system

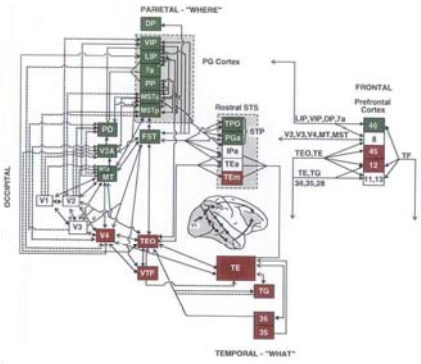
a), Understanding an image needs a vast amount of prior knowledge about the world !
Most of the objects cannot be detected by local appearance. For example, the functional object categories in PASCAL VOC benchmark have performance less than 20%

More top-down connections than bottom-up links

In the visual pathways, there are more downward (top-down) and lateral connections than forward (bottom-up) connections (10 :1)

Human visual pathways





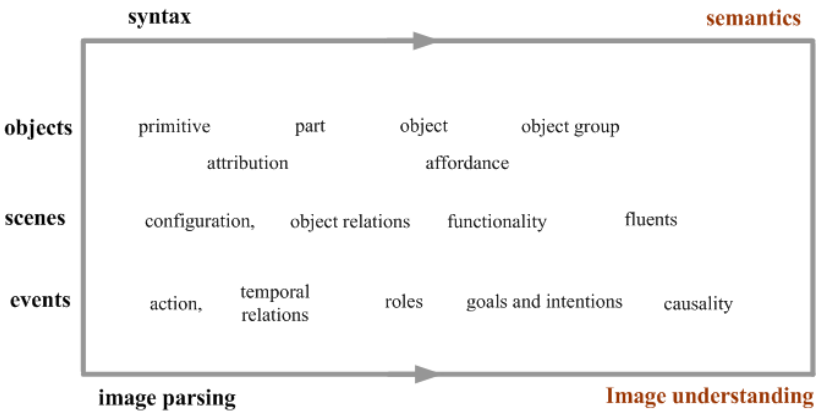
Observations of the vision system

b), Vision seems to be a continuous a computational process:
---- the more you look, the more you see.

Image shown to subjects	40ms	80ms	107ms	500ms
	"Possibly outdoor scene, maybe a farm. I could not tell for sure."	"There seem to be two people in the center of the scene."	" People playing rugby. Two persons in close contact, wrestling, on grass. Another man more distant. Goal in sight."	"Some kind of game or fight. Two groups of two men. One in the foreground was getting a fist in the face. Outdoors, because I see grass and maybe lines on the grass? That is why I think of a game, rough game though, more like rugby than football because they weren't in pads and helmets..."

Human subjects reporting on what he/she saw in an image shown for different presentation durations (PD=27, 40, 67, 80, 107, 500ms). from L. Fei-Fei and P. Perona 2007

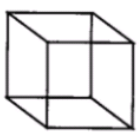
Vision is a continuous (literally infinite) computing process



Observations of the vision system

c), resolving local and global ambiguities.

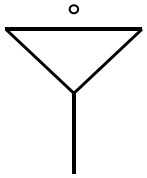
In mathematical terms, our perception can switch or jump in some structured state space.



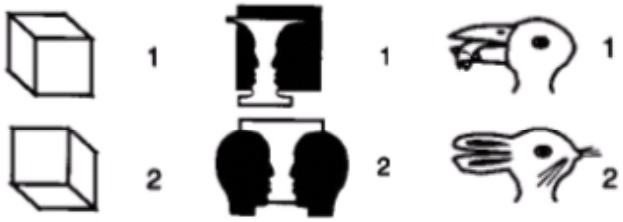
Necker cube

face / vase

rabbit / duck



bikini vs. martini



A common property is that the individual elements are strongly coupled and those strongly coupled elements must change their labels together. It is very hard to implement.

Here are two more challenging examples

global ambiguity

Can computers find and switch between these solutions ?

Bayesian View

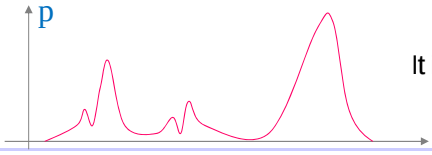
A basic assumption, since Helmholtz (1860), is that biologic and machine vision compute the most probable interpretation(s) from input images.

Let I be an image and W be a semantic representation of the world.

$$W^* = \arg\max_{w \in \Omega} p(W \mid I) = \arg\max_{w \in \Omega} p(I \mid W)\, p(W)$$

In statistics, we need to sample from the posterior.

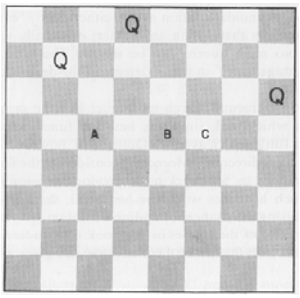
$$(W_1, W_2, \ldots, W_k) \sim p(W \mid I)$$



It is a sampling problem !

Ex 1: 8-Queen problem

Put 8 queens in a 8 x 8 chess board so that they are safe: i.e. no two queens occupy the same row, column, or diagonal lines.



Inference 1: 8-Queen problem

This is a *constraint-satisfaction* problem on a 8 x 8 grid.

Let's define s be a solution, s could be a binary 8x8 matrix or a list of the coordinates for the 8 queens.

Define the solution in a set:

$$\Omega* = \{\ s :\quad h_i(s) <= 1,\quad i=1,2,\ldots,46\ \}$$

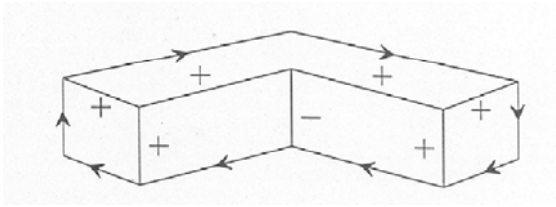
$h_i(s)$ is a hard (logic) constraints respectively for the 8 row, 8 column, 30 diagonal lines.

The computational problem is

$$find \qquad s \in \Omega *$$

Ex 2: Line drawing interpretation

Label the edges of a line drawing (graph) so that they are consistent



This is also *constraint-satisfaction* problem on a graph G=<V,E>.

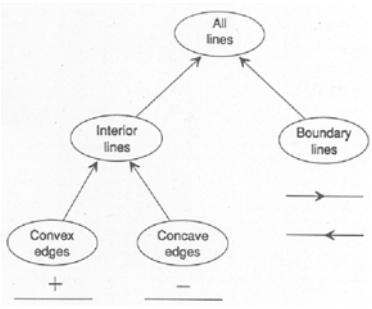
Define the solution in a set:

$$\Omega* = \{\ s :\quad h_i(s)=1,\quad i=1,2,\ldots,|V|\ \}$$

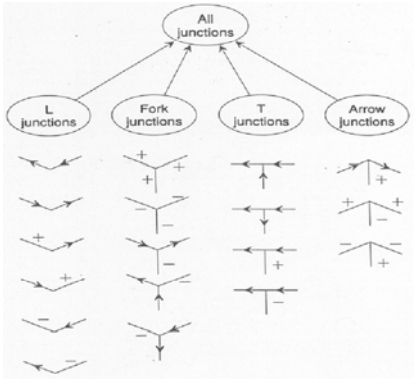
$h_i(s)$ is a hard (logic) constraints respectively for consistence at each vertex.

Ex 2: Line drawing interpretation

allowed edge labels



allowed junction labels



These allowed labels and junctions represent strong constraints and thus prior knowledge.

Ex 3: Channel codes

Binary-channel codes can be seen as a set of bits that must satisfy a number of constraints

$$g(x_1, x_2, \ldots x_n) \in \{ X: x_1 \otimes x_3 \otimes x_5 = 0; x_1 \otimes x_2 \otimes x_3 \otimes x_4 = 0 \}$$

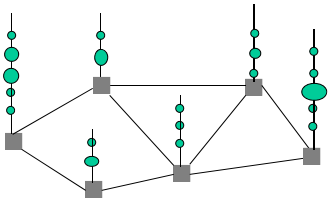
	x_1	x_2	x_3	x_4	x_5
c_1	1	0	1	0	1
c_2	1	1	1	1	0
c_3	0	1	1	1	1

Descriptive methods: summary

There are many more similar examples, e.g.
image restoration, image segmentation, graph partition/coloring, shape from stereo/motion/shading …

Common properties:
1. A graph representation G=<V, E>.
   G could be directed, undirected, such as chain, tree, DAG, lattice, etc.

2. hard constraints or soft "energy" preference between adjacent vertices.



Descriptive methods: summary

These problems belong to the descriptive family. The computing algorithms includes:
Relaxation-Labeling, Dynamic programming (I consider HMM as descriptive model not generative),
Belief propagation,
Gibbs sampler, Swendsen-Wang, Swendsen-Wang cut.

Issues in algorithm design:

1. Visiting scheme design and message passing.
   which step is more informative, relax more constraints (like line-drawing). In general, the ordering of Gibbs kernels
2. Computing joint solution or marginal belief.
   the marginal believe may be conflicting to each other.
3. Clustering strongly-coupled sub-graphs for effective moves.
   the Swendson-Wang ideas.
4. Computing multiple solutions.

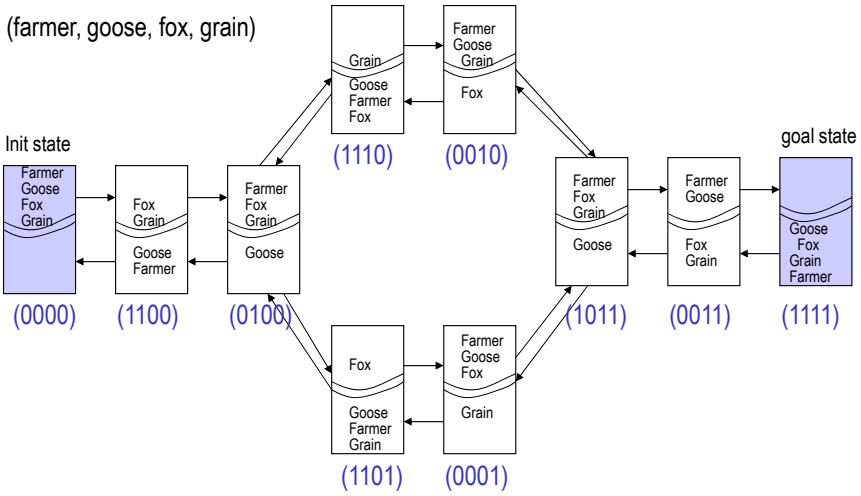
Ex 4: farmer, goose, fox, and grain

A *farmer* wants to move himself, a silver *fox*, a fat *goose*, and some Tasty *grain* across a river. Unfortunately, his *boat* is so tiny he can Take only one of his possessions across on any trip. Worse yet, an Unattended fox will eat a goose, and an unattended goose will eat Grain.

How can he cross the river without losing his possessions?

This can be formulated as finding a path in the state-space graph (next page).
In the coin example, the path is further extended to and-or graph.

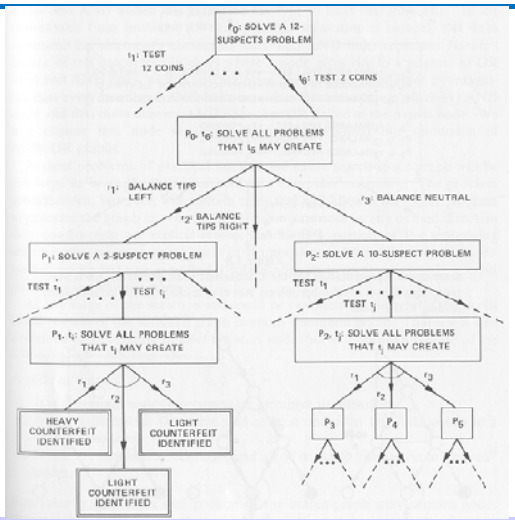
The State Space Graph



Ex 5: 12 Counterfeit coin problem

Given 12 coins, one is known to be heavier or lighter than the others. Find that coin with no more than 3 tests using a two-pan scale.

This generates the And-Or graph representation.



And-Or Graph is also called “hyper-graph”

The and-Or graph represents the decomposition of task into sub-tasks recursively.

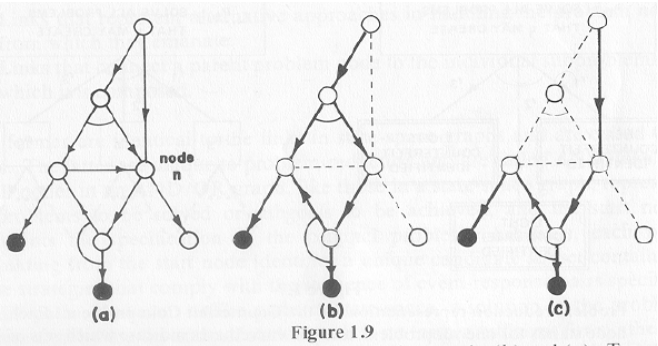
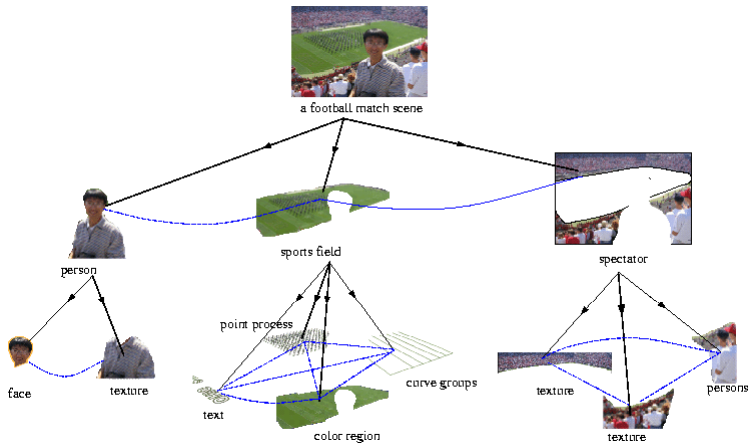


Figure 1.9
An AND/OR graph (*a*) and two of its solution graphs (*b*) and (*c*). Terminal nodes are marked as black dots.

Ex 6: Images parsing

Tu et al 2002-05

Parsing an image into its constituent visual patterns. The parsing graph below is a solution graph with AND-nodes



State space decomposition

A key concept in vision is composition that complex visual patterns, such as scene, objects are composed of simple elements. This leads to product state spaces. Anatomize the state space is a crucial aspect towards effective algorithm design.

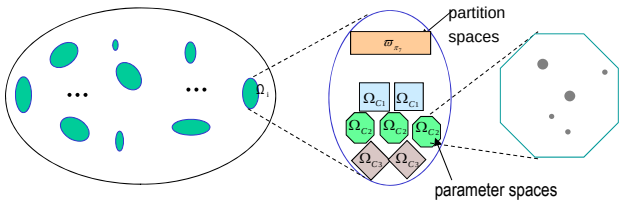
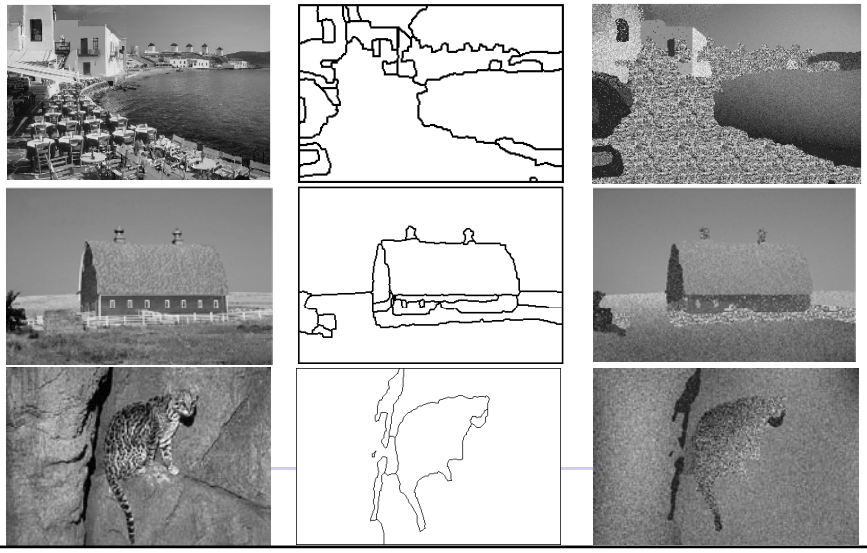


Image segmentation by Data Driven Markov Chain Monte Carlo

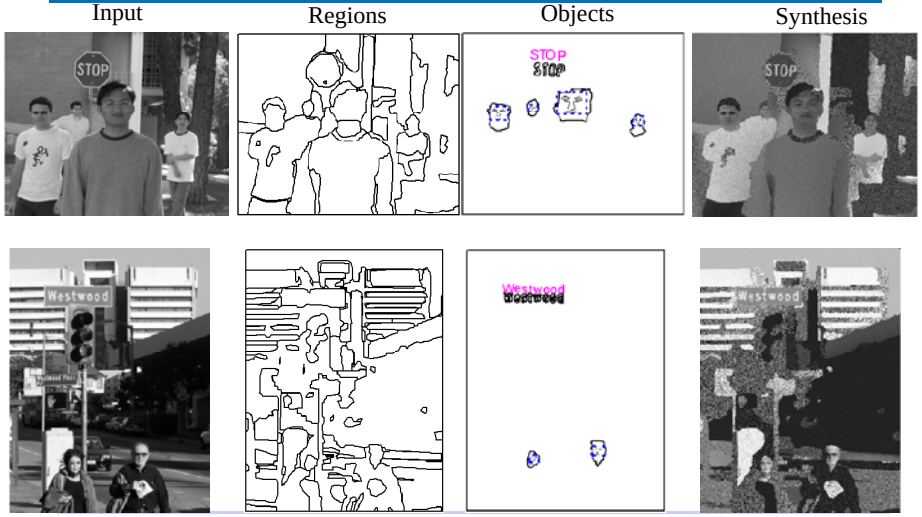
(Tu and Zhu, 01)

a. Input image | b. segmented regions | c. synthesis $I \sim p(I \mid W^*)$



Some Image Parsing results

Input | Regions | Objects | Synthesis

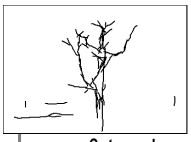


Tu, Chen, Yuille, and Zhu, iccv2003

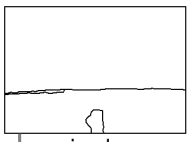
Example: 3D Sketch from a single image

Example II: 3D reconstruction (Han and Zhu, 2003)

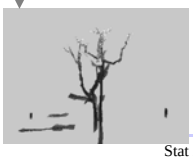
input I

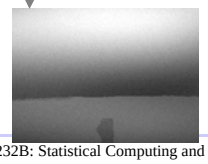


curve & tree layer



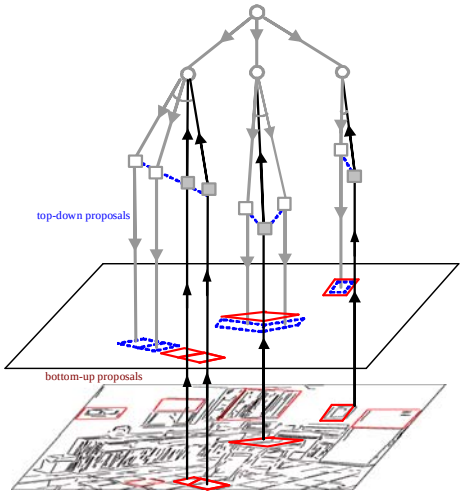
region layer

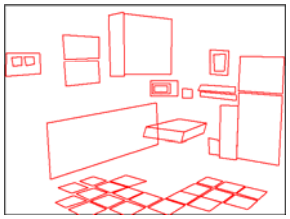




3D reconstruction and rendering

Top-down and Bottom-up Search





Integrating generative and discriminative methods

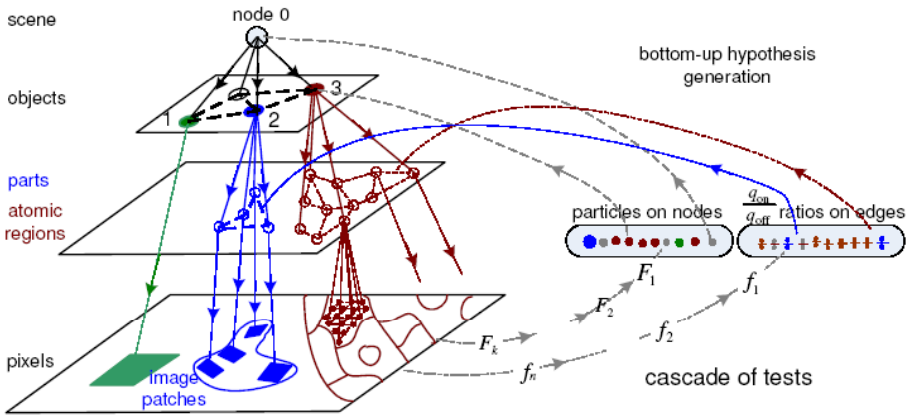
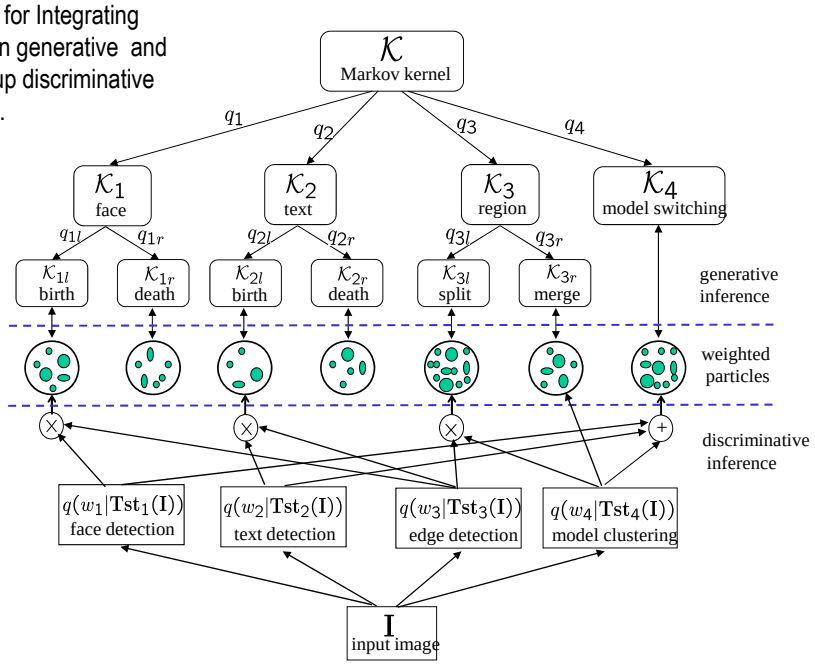


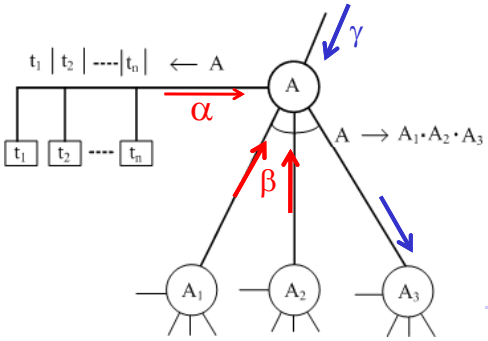
Diagram for Integrating Top-down generative and Bottom-up discriminative Methods.

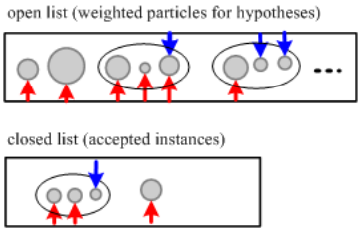


Recursive computing and parsing

In the And-Or graph --- a recursive structure.
we only need to consider a single node A.

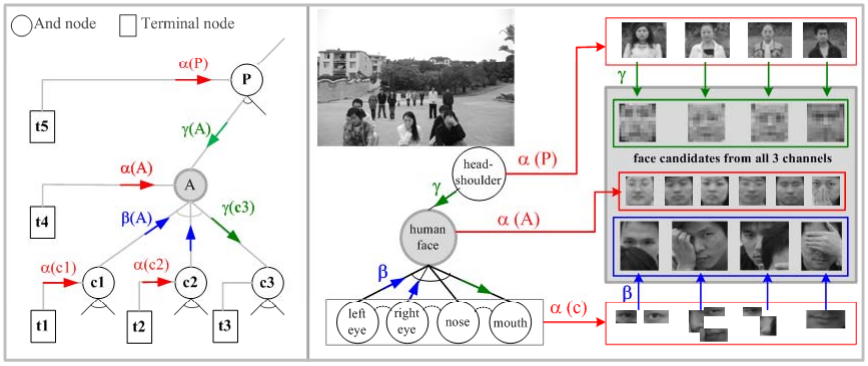
1, any node A terminate to leaf nodes at a coarse scale.
2, any node A is connected to the root.





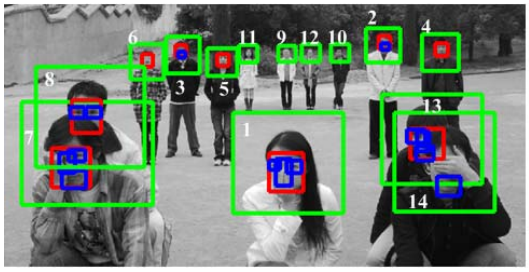
Compositional boosting, T.F. Wu et al, CVPR 07

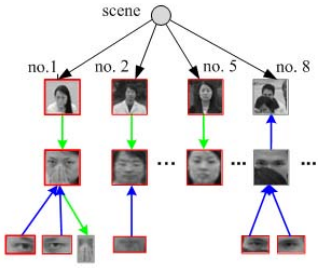
Recursive parsing: the α, β, γ -processes



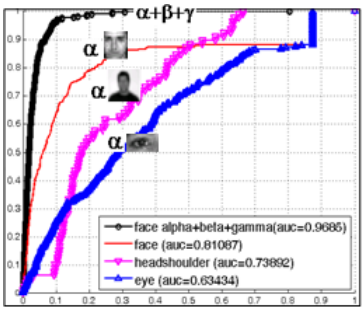
T.F. Wu et al 2009

Ordering the α, β, γ -processes

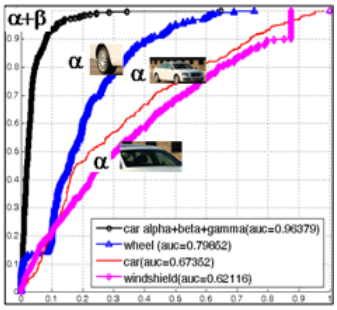




Ordering the α, β, γ -processes

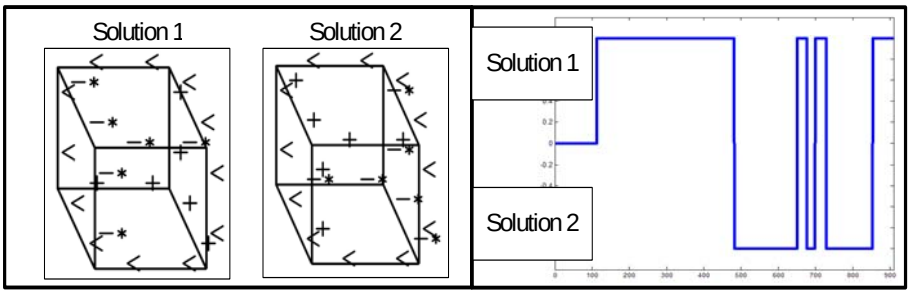


(a) head/shoulder---face---eyes



(b) car---parts

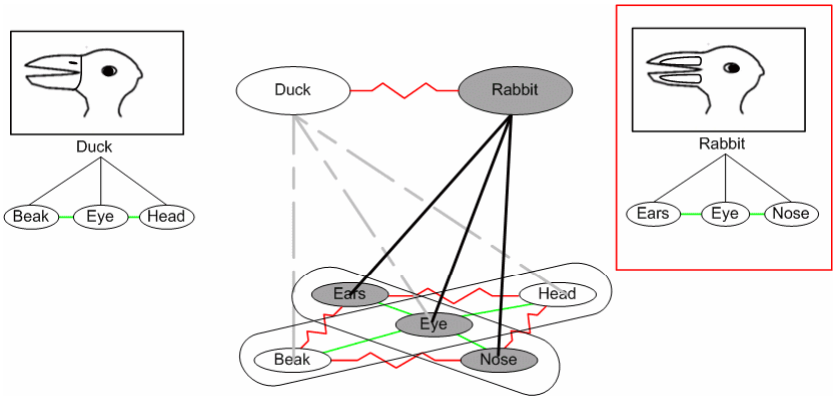
Solving ambiguities: the Necker Cube



From Porway et al 09

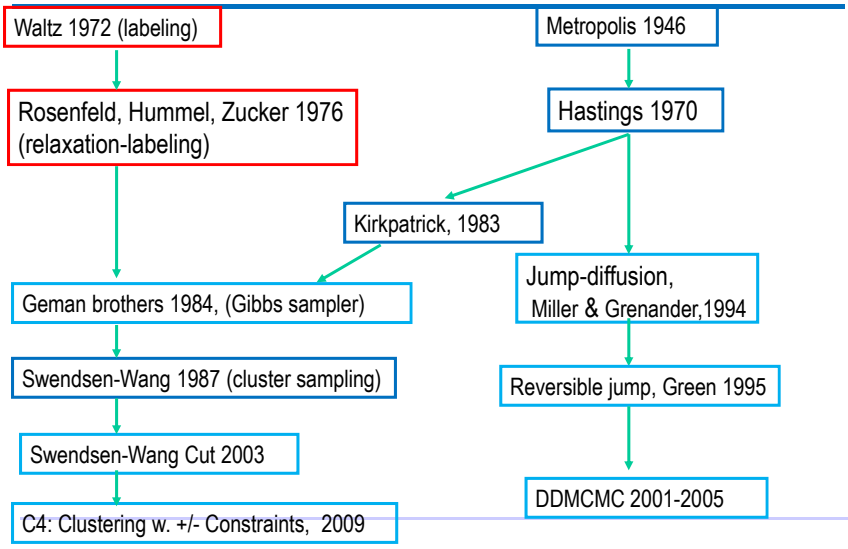
Solving ambiguities: the duck-rabbit ambiguity

The candidacy graph so far represent pair-wise edges, high-order relations are represented by extended candidacy graphs.



System will now flip between duck and rabbit without love triangle issue.

Summary of MCMC algorithms



More general setting

The computing framework , so far, has assumed that we are designing an algorithm to optimize a given function (energy in a Gibbs probability or a posterior probability).

In a more general setting, a system must work in two modes:

1, Exploitation (inference): using current imperfect model, make inference.
2, Exploration (learning): obtaining and updating the model.

A typical example is to play chess game. Actually any intelligence system should adopt this strategy. This will be covered as Bandit problem or Monte Carlo planning In this class.