A* Talk
A.L. Yuille (UCLA)
Three Examples of A*

- Example 1: Interactive Segmentation.
- Example 2: Road Tracking – Geman and Jedynak 1996.
- Example 3: A* for Hierarchical Object Models (Kokkinos).
Example 1: Interactive Segmentation.

• Graph is the Image lattice.
• Two points on the graph are specified – A & B.
• Find shortest path between A & B.
• Now go to hand-written notes.
Example 2: Road Tracking

• Inspired by Geman and Jedynak (1996).
• Find a road in an aerial photograph.
Road Tracking

- Example:
How to search for the road?

• There are an exponential number of possible paths.
• You do not have time to search them all.
• You must select a search strategy that is efficient.
• Geman and Jedynak proposed a new strategy based on information theory.
• Their strategy is to search so as to maximize the expected gain in information – see Jedynak’s talks on Wednesday.
• Coughlan and Yulle analyzed the algorithm – and showed that it was a variant of an inadmissible A* algorithm.
Third Example: Detecting Object in Image

• Hierarchical Models of Objects.
• A* over rules for combining subparts of objects to build a complete object.
• This is used to detect objects – cars – in images.

• Why hierarchical models? (more in week 3).
• More robust than “flat models” – can detect even if subparts of the object are missing (occluded or undetected).
• Ability to share parts (not used in this example).
Example 1: Interactive Segmentation

Image lattice

Set $X_A$ at $A$

Set $X_B$ at $B$

Path $X = X_A, X_1, \ldots, X_n, X_B$

where $X_i, X_{i+1}$ are neighbors on the graph $C$

Find best curve between $A$ and $B$.

Best Curve: Smoother (geometry)

Elginess - evidence for edges on curve.
Energy Cost. \[ E(X) = \frac{1}{2} \sum_{i=1}^{N} \varphi_i(X_i) + \sum_{i=1}^{N-1} \varphi_i(X_i, X_{i+1}) + \varphi_r(X_A, X_i) + \varphi_B(X_N, X_B). \]

- Unary + Binary terms (e.g. see Boykov)
- Binary term imposes geometry – e.g. the curve is smooth.
- Unary term imposes image term – e.g. the curve goes through pixels where there is evidence for an edge (e.g. \(|\nabla I|\) is big)

Task: Solve \[ \hat{X} = \arg \min_X E(X) \]
we cannot search exhaustively over all curves – too many. (exponential)
(iii) **A* Algorithm**.

Partial path \( X^t = (X_A, X_1, \ldots, X_t) \) has cost
\[
\frac{t}{2} \sum_{i=1}^{t} \phi_i(X_i) + \phi_A(X_A, X_1) + \sum_{i=1}^{t-1} \phi_i(X_i, X_{i+1})
\]
plus heuristic cost \( f(x^t, x^b) \), measured.

Apply A*.

**What heuristic cost?** Depends on form of \( \phi_i \)’s.

**Admissible heuristic**: An underestimate of cost to get from \( X_t \) to \( X_b \)
\[
\min_{\text{all paths}} \left\{ \frac{t}{2} \sum_{i=1}^{t} \phi_i(X_i, X_{i+1}) + \phi_T(X_t, X_b) \right\}
\]
(iv) In some cases, we can set $f(X_t, X_b) = 0$ if $\varphi \geq 0$.

E.g., Dijkstra's Algorithm (Geiger & Liu, 1996)

Set of partial curves at time $t$. $X_t$ can expand all nodes $X_b$.

Which node to expand?

Node which minimizes Cost of Partial Curve + Cost of Heuristic.

Guaranteed to converge to minimum cost path (if heuristic is admissible)
(1) Example 2: Find a road (highway) from Aerial Image

Model: road consists of segments.
(e.g. length 8 pixels)

Start of Road is given.

A road path is a sequence of N segments

\[ \text{e.g. } \overrightarrow{\text{or } \overrightarrow{\text{or }} } \]

There are an exponential no. of possible road paths

\[ \rightarrow 3^N \text{ road paths} \]
(2) Probability distribution over road paths.

representation: \( X = (x_1, \ldots, x_N) \)

position of first segment (specified)

\[ P(X_{t+1} | X_t) \rightarrow \begin{cases} \frac{1}{2} & \text{straight} \\ \frac{1}{4} & \text{left} \\ \frac{1}{4} & \text{right} \end{cases} \]

\[ P(X) = \prod_{t=1}^{N-1} P(X_{t+1} | X_t) \quad \text{Markov property} \]

Prob for the complete road path.
Image Model:

Set of all possible segments \( a \in A \).

Test \( Y_a \) - image filter applied to segment \( a \).

E.g., edge detail of filter (unimportant) or anything else.

\[
P(Y_{\text{on}} | a \text{ on road}) \quad P(Y_{\text{off}} | a \text{ off road})
\]

Local cues are ambiguous distribution overlap.
Bayes' Formula:

\[ P(Y|X) = \prod_{a \in A} P(y_a|X_a) \]
\[ P(X) = \prod_{t=1}^{T} P(x_{t+1}|x_t) \]

Assume that the image filter responses are independent.

Want to maximize

\[ P(X|Y) = P(Y|X)P(X) \propto P(Y|X)P(X) \]

Solve:

\[ X^* = \text{argmax}_{X} \frac{P(Y)}{P(X)} \] proportional to \[ P(Y|X) \]
(5) Cannot solve $\hat{X} = \text{arg max}_X \prod P(Y|X)P(X)$ by exhaustive search.

There are too many possibilities → $3^N$ road paths.
Can't evaluate all these paths, can't even store them.

$\hat{X} = \text{arg max}_X \left\{ \log P(Y|X) + \log P(X) \right\}$

or $\hat{X} = \text{arg max}_X \left\{ -\log P(Y|X) - \log P(X) \right\}$
(6) The form of $P(X)$ and $P(Y|X)$ enables us to use $A^*$. 

Note: this is like Mr. Korf's road search problem with GPS.

End node

Note: In G2S, there is no end node. So we pretend that we have one.

What is cost for each segment?

Two terms: (i) output of image filter: $- \log P(Ya|a \text{ on road})$. 

(ii) geometry (depends on previous node): $P(Ya|a \text{ off road})$. 

15% $\rightarrow$ $\log P(X^{t+1}|X^t)$. $X^{t+1}$ is segment $a$. 

or $\rightarrow$ $\cdots$.
(7) To apply A* we need a heuristic.

Score: \[ \frac{1}{t=1} \sum_{t=1}^{T} \log P(X_{t+1}|X_{t}) - \frac{1}{a=1} \sum_{a=1}^{T} \log \frac{P(Y_{a}|a \text{ on road})}{P(Y_{a}|a \text{ off road})} \]

Heuristic is lower bound of the rest of the path.

Need to lower bound \[ \min \left\{ -\frac{1}{T+t} \sum_{t=T+1}^{T} \log P(X_{t+1}|X_{t}) \right\} \]

\[ -\frac{1}{a=T+1} \sum_{a=T+1}^{T} \log \frac{P(Y_{a}|a \text{ on road})}{P(Y_{a}|a \text{ off road})} \]
Special Case.

One idea: let the heuristic be \((N - M) C\), where \(N\) is the total length of the road, \(M\) is the no. of segments we have travelled so far, and \(C\) is a constant. \(\leq\) cost so far.

Our cost is \[-\sum_{t=1}^{M} \left( \log P(X^{t+1}|X^t) + \log P(Y^t|X^t) \right) + (N-M)C \leq \text{heuristic cost}\]

\[= -\sum_{t=1}^{M} \left( \log P(X^{t+1}|X^t) + \log P(Y^t|X^t) + C \right) + NC \leq \text{Note: independent of path and of \(M\).}\]
(9) This choice of heuristic means that we do not need to have an end node; i.e., drop the NC term. We can continue for ever.

Intuition: expand a road path by adding a segment, compute its extra cost: \(- \log P(x^n|x^M) - \log (y^n|x^n)\). Subtract \(c\)

Path AEF is longer than path AE (by one segment). But subtracting \(c\) penalizes the length.

Hence
What choice of heuristic?

If admissible, then \(-c \leq \min_{x^t, y^t} \left( -\log P(x^{t+1}|x^t) - \log P(y^t|x^t) \right)\).

But, this is bad for this example.

Why, because it means we are not satisfied with any segment.

Expand this segment, measure \(-c\) as \(-\log P(x^{t+1}|x^t) - \log P(y^t|x^t)\). Add heuristic \(-c\), get result \(\leq 0\).

So do not expand A further. Instead expand B, (same result) then expand D.
Problem with this heuristic \( \Rightarrow \) reduces to breadth-first search.

Intuition \( \Rightarrow \) we are never satisfied with the local maximum:

\[- \log P(x_{t+1} | x_t) - \log P(x_{t+1} | x_t) \] because it is worse than \( C \).

So we go back and expand an earlier path.

\( \ell \) not good enough

\( \ell \) not good enough

\( \ast \) Expand \( \ell \) again.

This requires searching \( 3^n \) paths.

Exponential \( \Rightarrow \) impossible
This is an unusual case \( \rightarrow \) Not standard A*

What to do?

- Inadmissible Heuristic.
  Cannot guarantee convergence to best solution
  But can make statistical guarantees (Coughlin and Yinill)
- With high probability we can find a path
  which is close to the true path in \( O(N) \) time

Like Probably Approximately Correct (PAC) Theorem in Machine Learning
(Vapnik/Valiant, McAlester)
How to do this? 

**Brief Sketch.**

Problem formulation assumes probability distributions $P(X)$ and $P(Y|X)$.

Can use these distributions to analyze the algorithm.

Start:

```
partial path $X_t$ → An admissible heuristic gives a lower bound of cost from $X_t$ to $X_b$.
```

Instead, specify a heuristic cost for each segment (not a lower bound) and compute the probability that the algorithm wastes time exploring false paths.
Some intuition. \[
\text{cost of segment } \leq \frac{1}{3},
\]

There is a probability \[P(3 \text{ on road})\]
probably \[P(3 \text{ off road})\]
specified by \[P(x), P(y|x)\]
specified by \[-\log P(3 | \text{on}) - \log P(1)\]
\[
P(3 | \text{off})
\]
Admissible Heuristic is \[\leq (N-M) \min \{P(3)\}\]

But it is very unlikely that we will have this cost.

If \(n - m\) is large, law of large numbers says that it will be closer to \[(N-M) \langle P(3) \rangle\] expected value.
Example 3

I. Kokkinos
Hierarchical Compositional Models

- **Top-down view**: object generates tokens
- **Bottom-up view**: object is composed from tokens
Compositional Detection

- View production rules as composition rules
  \[(p_{p_1}, \ldots, p_{p_n}) \rightarrow p_O\]

- Build a parse tree for the object

- Requires
  - Composition rules
  - Prioritized search
Composition of the `Back’ Structure
Composition as Climbing a Lattice

• Introduce vector indicating instantiated substructures

\[ I(S) = [1, 0, 1], \quad S = (S_1, -, S_3) \]

– partial ordering among structures

\[ S^i \preceq S^j \iff I_k(S^i) \leq I_k(S^j) \quad \forall k \]

• Hasse Diagram for 3-partite structure

– By acquiring a substructure, the structure climbs upwards
Composition of the `Back’ Structure

Problem: Too many options!
(Combinatorial explosion)
Analogy: Building a puzzle

• Bottom-Up solution: Combine pieces until you build the car
  – Does not exploit the box’ cover

• Top-Down solution: Try fitting each piece to the box’ cover.
  – Most pieces are uniform/irrelevant

• Bottom-Up/Top-Down solution:
  – Form car-like structures, but use cover to suggest combinations.
Best First Search

- **Dijkstra’s Algorithm**
  - Prioritize based on `cost so far`
  - For parsing: Knuth’s Lightest Derivation
- **A* Search**
  - Consider `cost to go`
  - Approximate with **heuristic** cost

![Labyrinth Diagram]

- Entry
- Exit
- Cost so far
- Cost to go
- Heuristic cost
`Cost to go’ for Parsing

- The Generalized A* Architecture, Felzenszwalb & McAllester
- Context: complement needed to get to the goal.

- Recursive derivation of contexts.

\[
\begin{align*}
\text{CON}(goal) &= 0 \\
(S_1 = w_1, S_2 = w_2) &\rightarrow (S_3 = w_3) \\
(S_1 = w_1, S_2 = w_2, S_3 = w_3, \text{CON}(S_3) = w_c) &\rightarrow \\
&\quad (\text{CON}(S_1) = w_3 + w_c - w_1) \\
&\quad (\text{CON}(S_2) = w_3 + w_c - w_2)
\end{align*}
\]
Heuristics for Parsing: Context Abstractions

• A* requires lower bound of derivation cost
• Derive context in coarser domain (abstraction)
  – Lower bound cost on fine domain

\[
\text{Cost} (\text{CON}(\text{Abs}(S_3))) \leq \text{Cost} (\text{CON}(S_3))
\]

• Use it to prioritize search

\[\text{KLD}: \ (S_1 = w_1, S_2 = w_2) \rightarrow_{w_3} (S_3 = w_3)\]

\[\text{A*} : \ (S_1 = w_1, S_2 = w_2, \text{CON}(\text{Abs}(S_3)) = w_h) \rightarrow_{w_3+w_h} (S_3 = w_3)\]
Abstractions via Structure Coarsening

- Coarsening: identify nodes of Hasse diagram

1 part suffices

- Lower bound composition cost

\[
\sum_{p \in P} \log P_{p|O}(p_{p|O}) = \sum_{p \in P} \frac{1}{2} \left[ \log((2\pi)^n |\Sigma_{p,O}|) + p_{p|O}^T \Sigma_{p,O}^{-1} p_{p|O} \right] \\
\geq \frac{1}{2} \left[ \log((2\pi)^n |\Sigma_{a,O}|) + p_{a|O}^T \Sigma_{a,O}^{-1} p_{a|O} \right] + \sum_{p \in P \setminus a} \max \left( \frac{1}{2} \log((2\pi)^n |\Sigma_{p,O}|), C_a \right)
\]
Coarse Level Parsing

KLD: Coarse Domain

Contexts to Fine Level

Bottom-Up

Top-Down
Fine Level Parsing

Top-Down Guidance: Heuristic, Coarse Level

Bottom-Up Composition, Fine level
A* versus Best First Parsing

- **A* Parsing**

  - Front Part
  - Middle Part
  - Back Part
  - Object
  - Goal

- **Knuth’s Lightest Derivation Parsing**
Parsing & Localization Results - I
Object – ETHZ Shape dataset
Parsing and localization results
ETHZ Benchmark results

UIUC cars

Apple logos

Bottles

Giraffes

Mugs

Swans

Recall vs. 1-Precision for different benchmarks and methods.
Forward pointers

- Learning the model parts:
  - Statistical Shape Models, 3rd week

- Learning the model parameters:
  - Latent SVM training, 3rd week

- Branch & Bound for star-shaped models:
  - Rapid Object Detection with Branch & Bound, 3rd week
    - spatial coarsening
    - score bounding