Static Analysis of Binary Executables Using Structural SVMs

Nikos Karampatziakis
Department of Computer Science Cornell University
Cornell University

30 Second Overview

- A structural SVM for solving sequence problems such as
  - *Finding basic blocks of code in binary executables (our task)*
  - Some kinds of scheduling problems
- What's so special about these problems?
- A linear time inference algorithm
- Two appropriate loss functions allowing fast training (and why Hamming loss is inappropriate)
- Experiments comparing against SVM^{hmm}, a discriminatively trained HMM

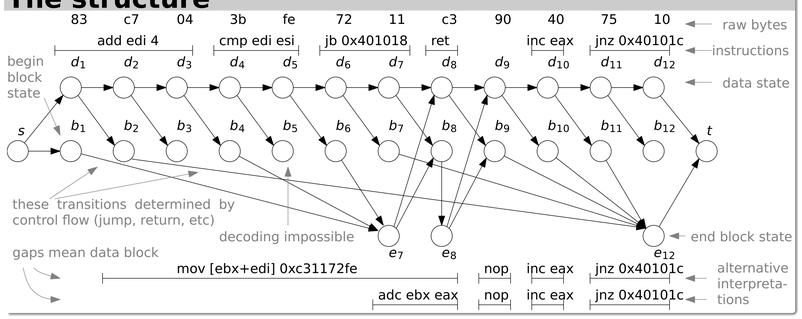
Our Task

- Segment binary executable into blocks of code (the actual instructions), and data (everything else), without running the program.
- Instructions have *no demarcations*.
  - Silimar problem with URLs such as `www.whorepresents.com`
  - Similar with 我们要学生活得有意义

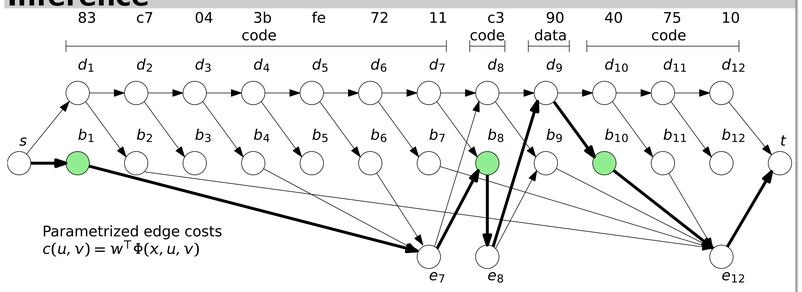
Isn't This Just Sequence Labeling/Segmentation?

- It's a *weighted interval scheduling problem*
- Data blocks cannot start in arbitrary positions
- Code blocks can be *really* long
- Given a starting position, the span of a code block is deterministic.
- Hamming loss inappropriate. Text analogy: parsing "driverballetterrace" as "<junk>, verbal, letter, race". Good overlap, but misses all words.

The structure

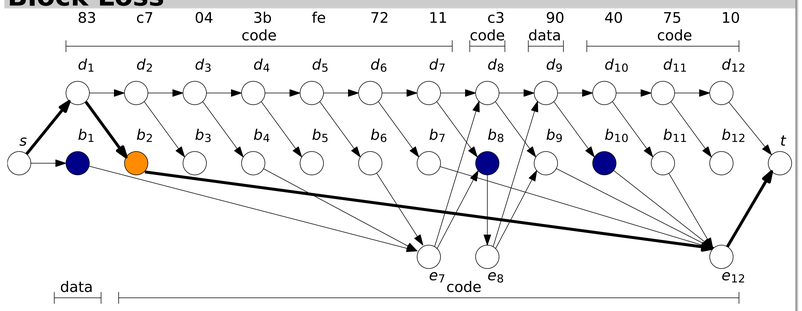


Inference



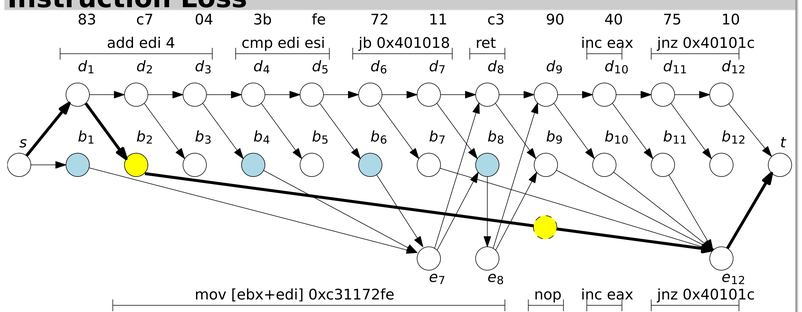
Heaviest *s*-*t* path in DAG. Linear time, dynamic program.

Block Loss



$\Delta_B(y, \bar{y}) = 4$ ($\bar{y}$ misses 3 (blue), uses 1 extra (orange))

Instruction Loss



$\Delta_I(y, \bar{y}) = 6$ ($\bar{y}$ misses 4 (light blue), uses 2 extra (yellow))

Mandatory Panel: SVM Formulation

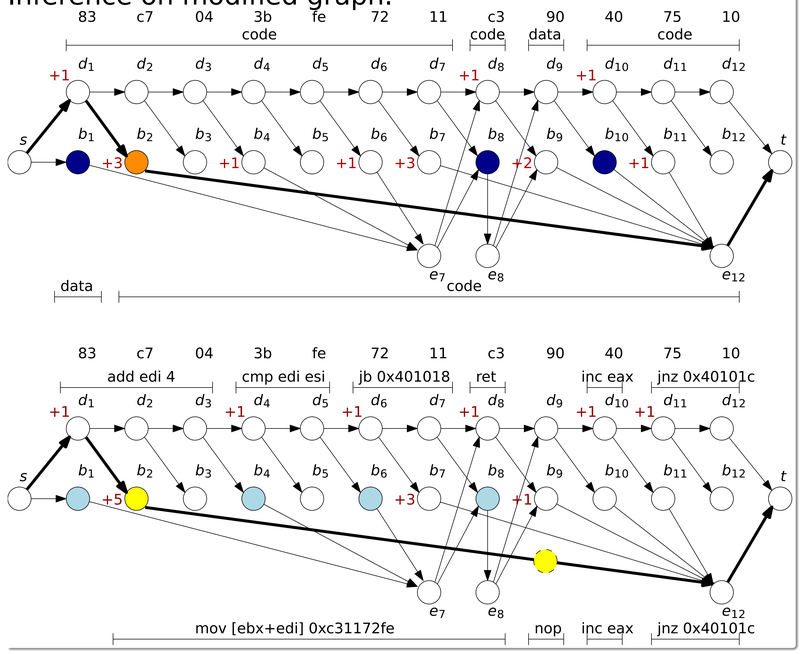
$$\min_{w,\xi} . \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \xi_i$$

$$\text{s.t } \forall i \; \forall \bar{y} \in \mathcal{Y}_i : \; w^\top \Psi(x_i, y_i) - w^\top \Psi(x_i, \bar{y}) \geq \Delta(y_i, \bar{y}) - \xi_i$$

where $\Psi(x, y) = \sum_{(u,v)\in y} \Phi(x, u, v)$
Solved with cutting plane algorithm.

Loss Augmented Inference (Constraint Generation)

Find $\arg\max_{\bar{y}} w^\top \Psi(x_i, \bar{y}) + \Delta(y_i, \bar{y})$
Fast: loss functions decompose over the vertices.
Inference on modified graph.



Experiments

200 binary executables from a typical Windows machine.

	Bytes	Blocks	Block length
Maximum	49152	3502	2794 bytes / 1009 instructions
Average	16712	887	13 bytes / 4 instructions

Features: unigrams and bigrams around each position, histograms of instructions for long edges
Δ_H: Hamming loss for reference
Δ_{NX}: normalized version $\Delta_{NX}(y, \bar{y}) = \frac{\Delta_X(y,\bar{y})}{|y|}$
$\bar{L}, \bar{I}, \bar{B}$: average length, instructions, blocks

	Δ_H	$\bar{L} \cdot \Delta_{NH}$	Δ_I	$\bar{I} \cdot \Delta_{NI}$	Δ_B	$\bar{B} \cdot \Delta_{NB}$
Greedy	1623.6	1916.6	2164.3	7045.2	1564.9	4747.2
SVM^{hmm}	236.2	201.3	—	—	45.1	46.9
SVM^{wis} Δ_I	98.8	115.6	44.6	98.0	26.1	41.1
SVM^{wis} Δ_{NI}	104.3	103.7	45.5	79.7	30.5	35.5
SVM^{wis} Δ_B	86.5	98.2	**39.6**	80.2	**21.5**	32.1
SVM^{wis} Δ_{NB}	**85.2**	**87.2**	40.6	**75.4**	23.4	**29.8**

Code & data available at www.cs.cornell.edu/~nk/svmwis

Email: nk@cs.cornell.edu
WWW: http://www.cs.cornell.edu