DFRWS
DIGITAL FORENSIC RESEARCH CONFERENCE

Automatic Reassembly of Document Fragments via Data Compression

By

Kulesh Shanmugasundaram

Presented At

The Digital Forensic Research Conference

DFRWS 2002 USA Syracuse, NY (Aug 6th - 9th)

DFRWS is dedicated to the sharing of knowledge and ideas about digital forensics research. Ever since it organized the first open workshop devoted to digital forensics in 2001, DFRWS continues to bring academics and practitioners together in an informal environment. As a non-profit, volunteer organization, DFRWS sponsors technical working groups, annual conferences and challenges to help drive the direction of research and development.

http:/dfrws.org

Automated Reassembly of Document Fragments

DFRWS 2002

Outline

- Introduction & Motivation
- Stages in Reassembly
- Reassembly Problem
- Our Solution
- Implementation & Experiments
- Summary

Introduction & Motivation

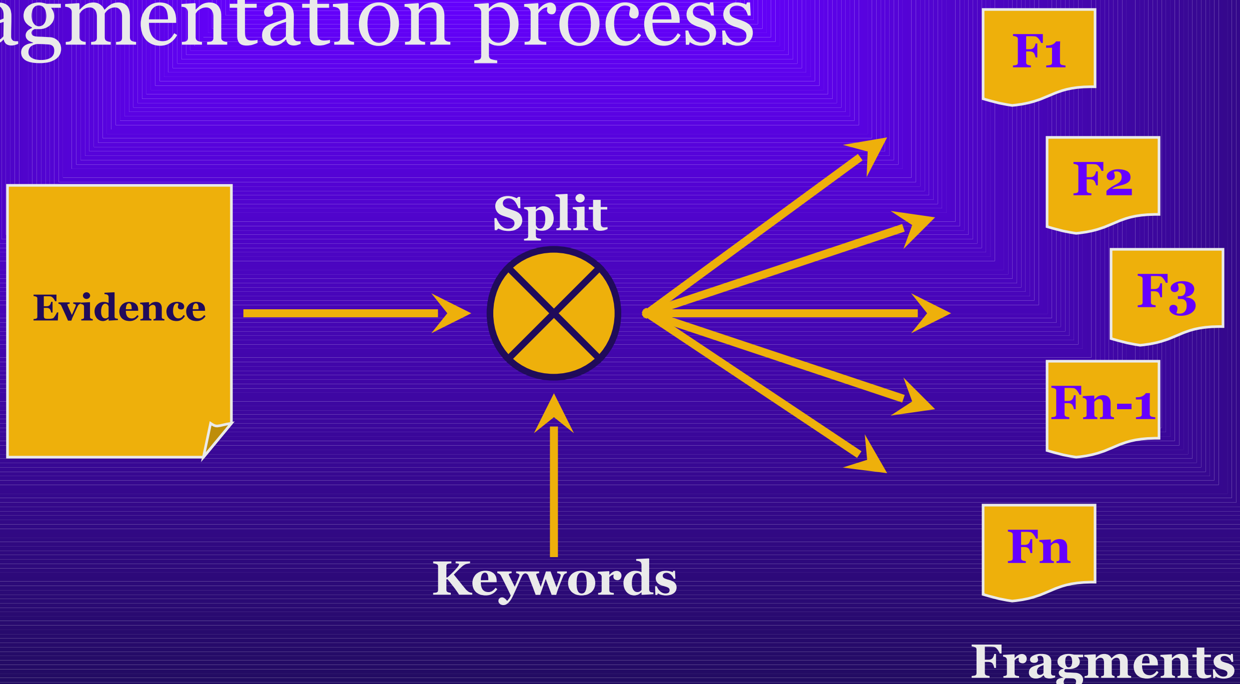
Introduction

- Reassembly of objects from mixed fragments
- Common problem in:
  - Classical Forensics
  - Failure Analysis
  - Archaeology

- Well studied, automated...
- Is there a similar problem in digital forensics?

Motivation

- Digital evidence is
  - malleable
  - easily scattered
- Fragmentation process

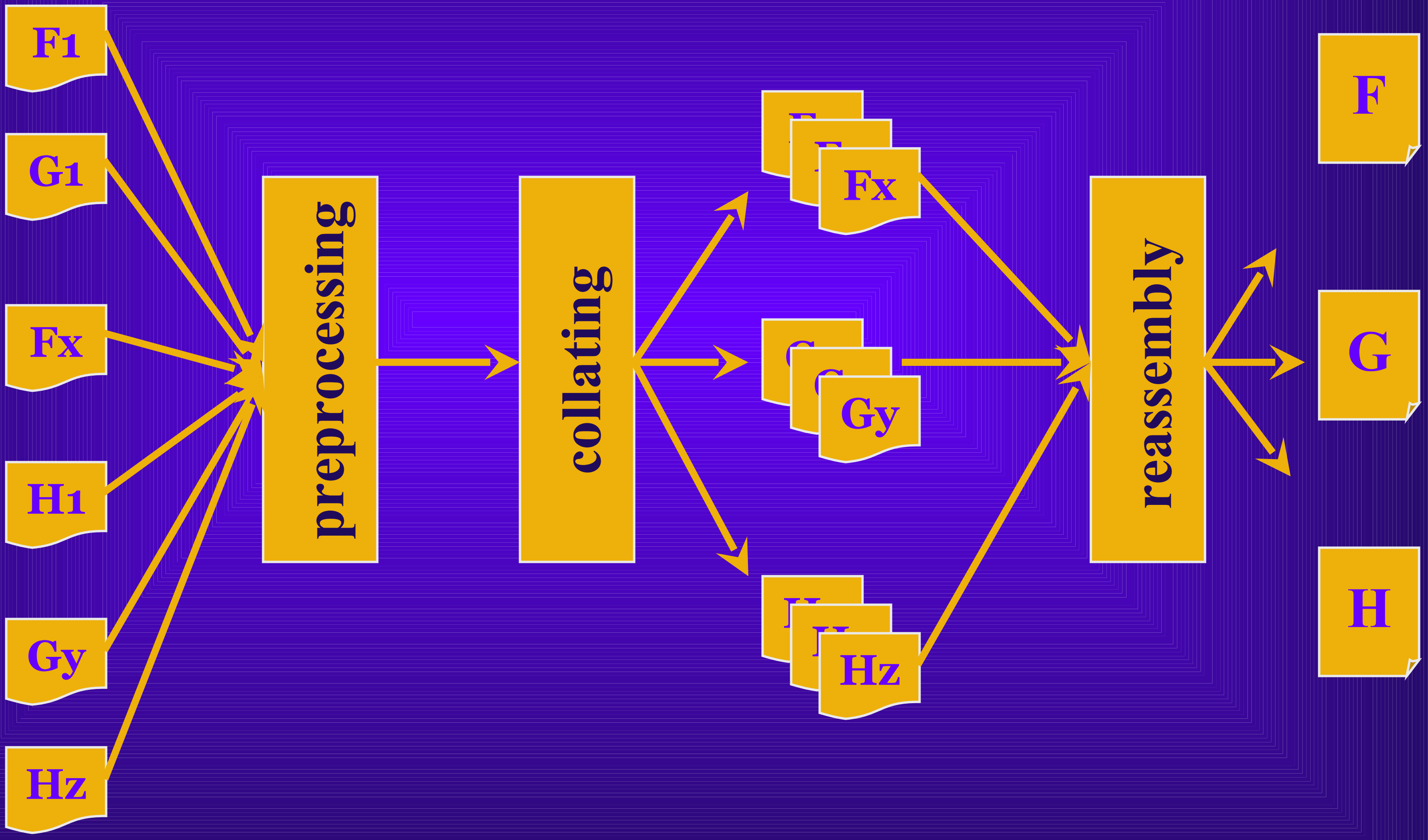


Motivation…

- Scenarios…
- Hiding in Slack Space
  - Criminal splits the document and hides them selectively into slack spaces based on a password
- Swap File
  - Addressing & state information is not available on the disk
- Peer-to-peer systems
  - Fragments are assigned a sequence of keywords and scattered across the network
  - e.g.: FreeNet, M-o-o-t

Stages of Reassembly

Stages of Reassembly



Stages of Reassembly

- Preprocessing
  - Cryptanalysis
  - Weight Assignments
- Collating
  - Group together fragments of a document
  - Hierarchical approach
- Reassembly
  - Reordering the fragments to form the original document

Reassembly

The Problem of Reassembly

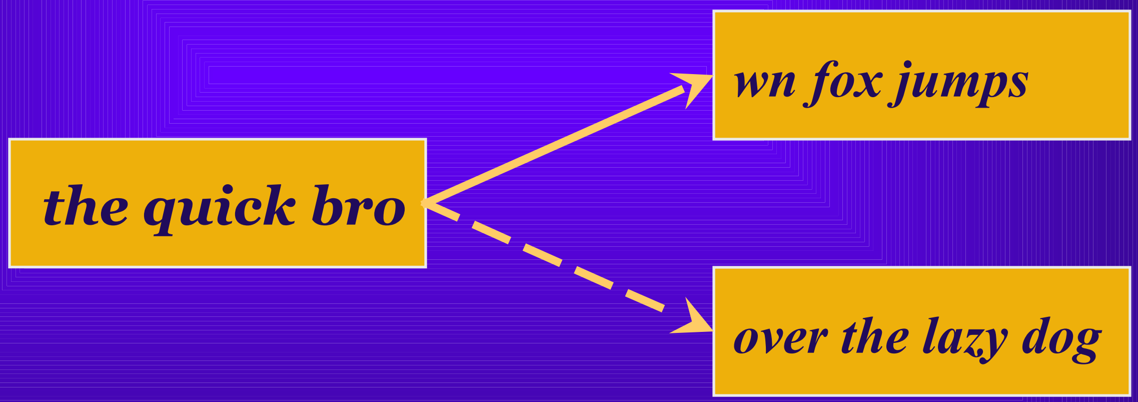
- Suppose we have fragments $\{A_0, A_1, \ldots A_n\}$ of document **A**
- Compute a permutation **X** such that

$$A = A_{X(0)} || A_{X(1)} || \ldots A_{X(n)}$$

- To compute A, we need to find adjacent fragments
- To reassemble:
  - Need to find adjacent fragments
  - Automate the process

Quantifying Adjacency

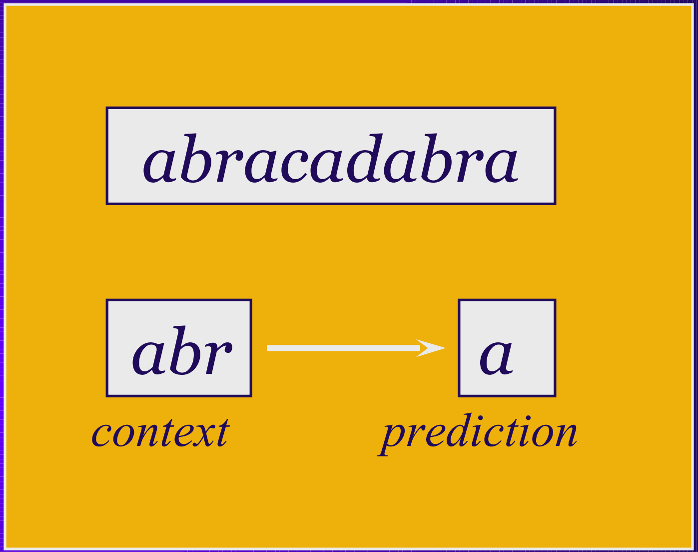
- An Example: A linguist may assign probabilities based on syntactic and semantic analysis



- This process is language dependent

Context-Based Statistical Models

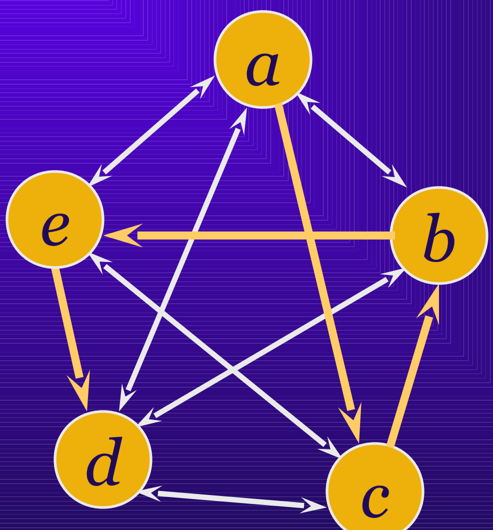
- Context based models are used in data compression
- Predicts subsequent symbols based on current context
- Works well on natural languages as well as other data types
- Context models can be used to predict upcoming symbols and assign candidate probabilities



Adjacency Matrix

- Candidate probabilities of each pair of fragments form complete graph
- A Hamiltonian path that maximizes the sum of candidate probabilities is our solution
- But this problem is intractable
- We will discuss a near optimal solution

	a	b	c	d	e
a	0	$c_{(a,b)}$	….		
b	$c_{(b,a)}$	0	….		
c					
d					
e					$c_{(e,a)}$



Steps in Reassembling

1. Build context model using all the fragments
2. Compute candidate probabilities for each pair
3. Find a Hamiltonian Path that maximizes the sum of candidate probabilities

Implementation & Experiments

Prediction by Partial Matching (PPM)

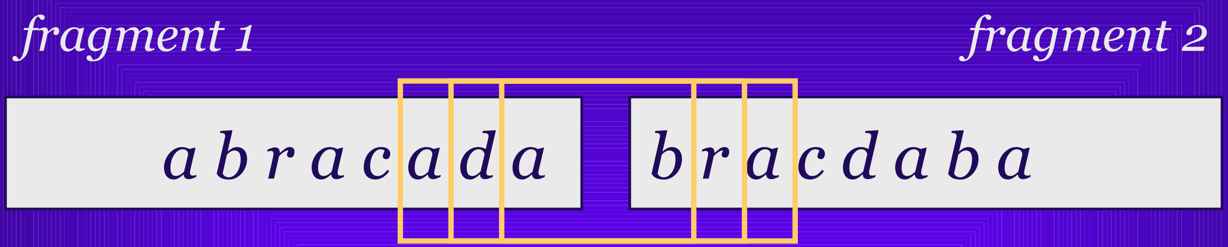
- Uses a suite of fixed order context models
- Uses one or more orders to predict upcoming symbol

- We process each fragment with PPM
- Combine the statistics to form a model for all the fragments

abracadabra

order=2	*order=1*	*order=0*
ab → r 2 2/3 → ^ 1 1/3	a → b 2 2/7 → c 1 1/7 → d 1 1/7 → ^ 3 3/7	→ a 5 5/12 → b 2 2/16 → c 1 1/16
ac → a 1 ½ → ^ 1 ½		
ad → a 1 ½ → ^ 1 ½		
br → a 2 2/3 → ^ 1 1/3		
ca → d 1 ½ → ^ 1 ½		

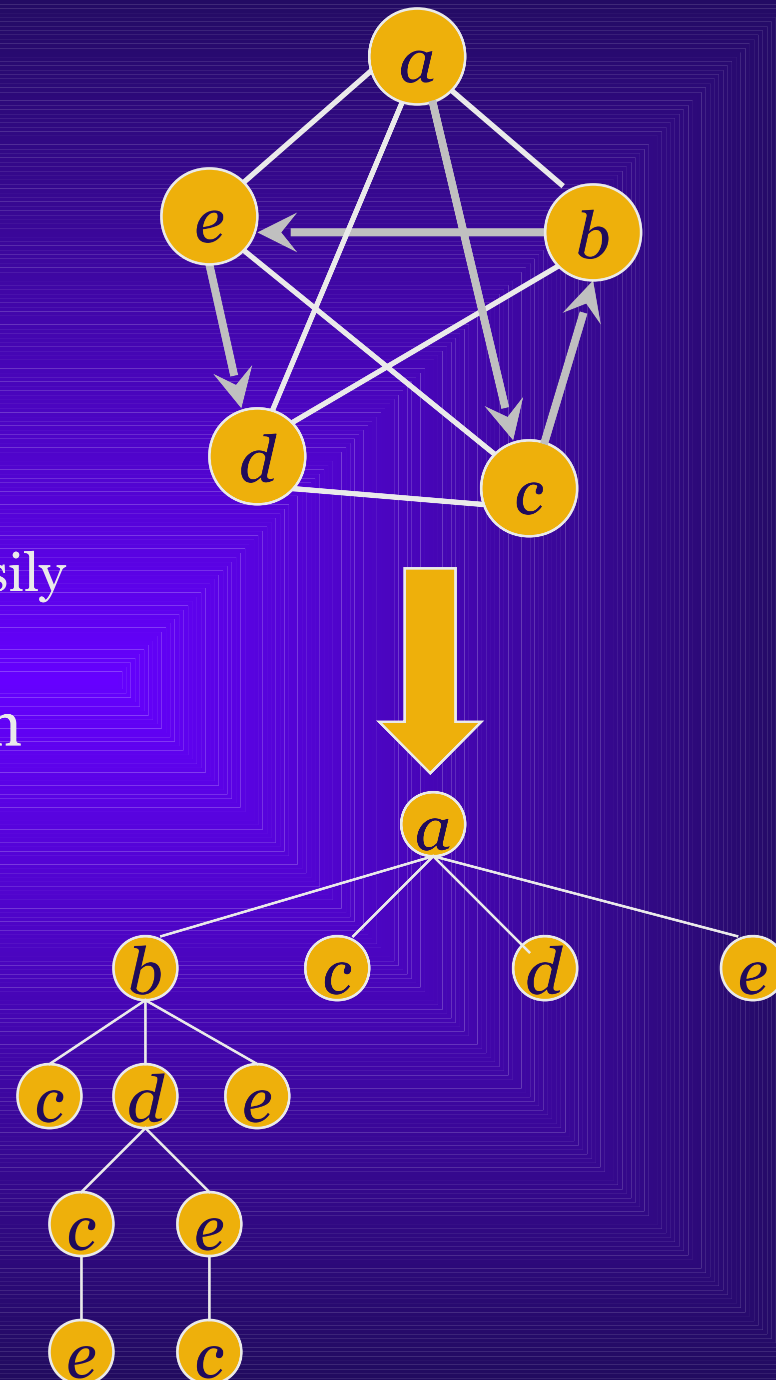
Candidate Probability



- Slide a window of size *d* from one fragment into the other
- At each position, use the window as context and determine the probability (p_i) of next symbol
- Candidate prob. $C_{(1,2)} = (p_o * p_1 * \ldots p_d)$

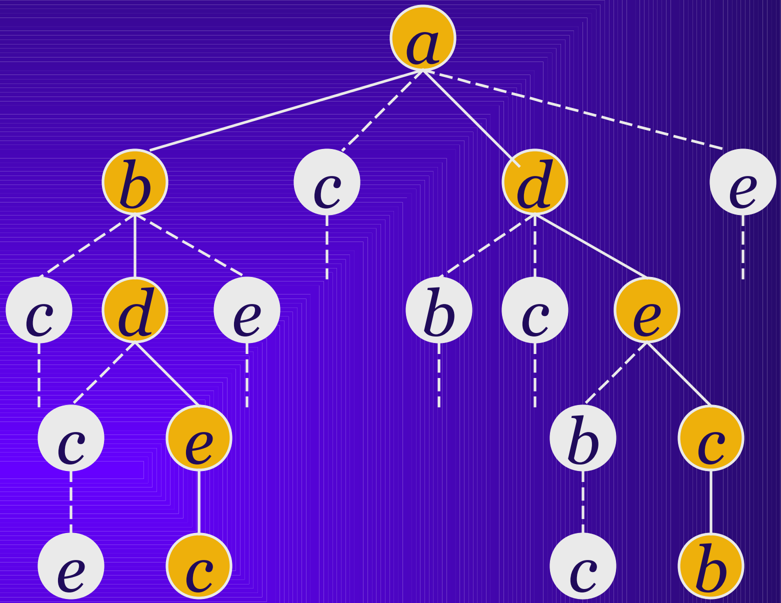
Solution Tree

- Assumtions:
  - Fragments are recovered without data loss
  - First fragment is known/easily identified
- Paths in complete path can be represented as a tree
- Tree grows exponentially!
- We have to prune the tree



Pruning

- At every level choose a node with the largest candidate probability
- We can choose *alpha* nodes at each level
- By looking at candidate probabilities *beta* levels deep

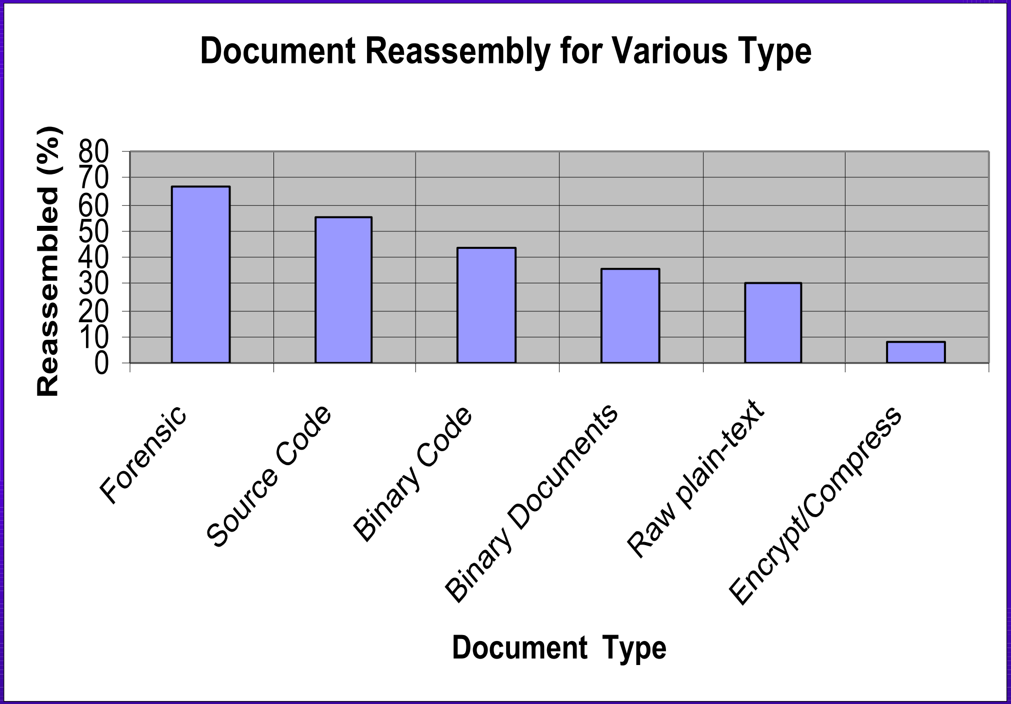


Experiments

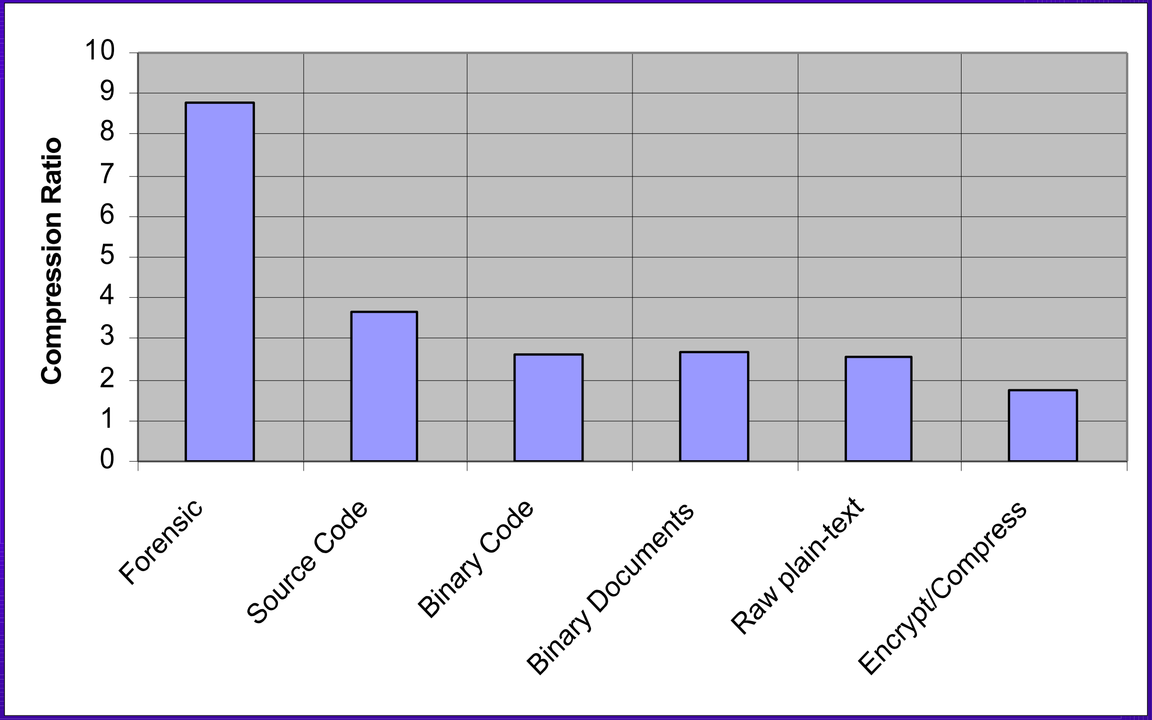
Data Set

OS Forensics	Log, history files of various users
Source Code	C/C++, Java source code
Binary Code	Executable, object code (Window, Linux, Solaris)
Binary Document	MS Office, PDF documents
Raw Plain-Text	Unformatted text, transcripts
Encrypted & Compressed	Encrypted, compressed files

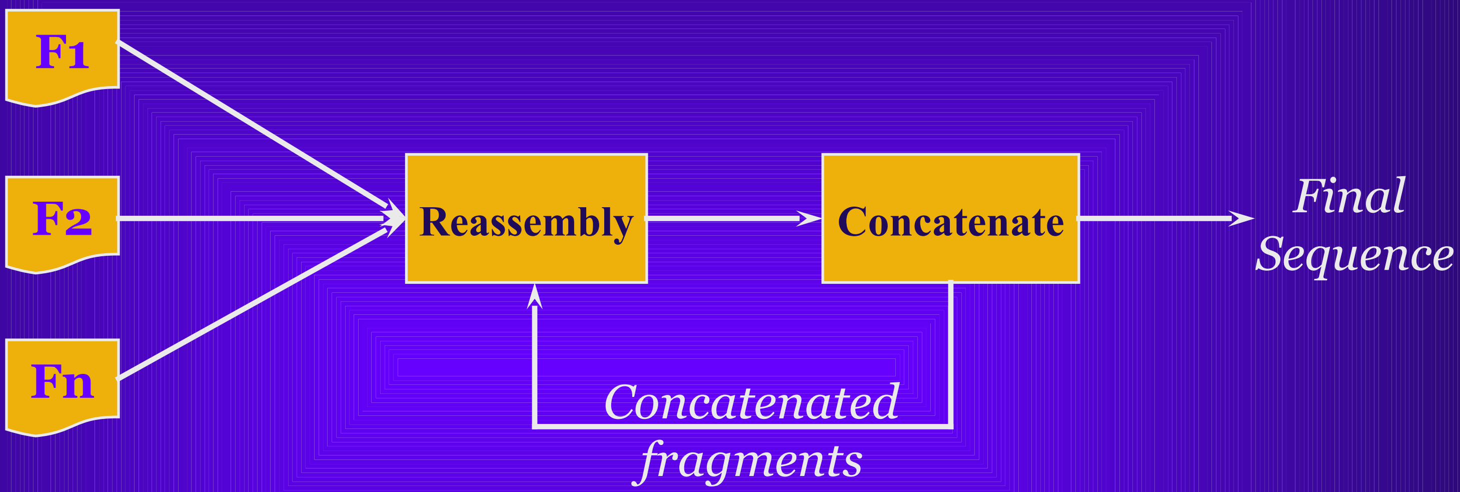
Reassembly for various types



Compression ratio



Iterative Approach



OS Forensic	4
Source Code	6
Binary Code	9
Binary Doc	10
Raw-text	10

Summary

- Introduced reassembly of scattered evidence
- Experiments & results
- Future work:
  - Identifying preprocessing heuristics
  - Compare performance with other models
  - Work on reassembling images

Questions