Fundamentals of Scientific Data

IST400/600
Jian Qin

Data Basics
at the low level of computing
Do you know…

- Bit?
- Byte?
- How do they form “word”?

Numbers

- Integer
  - Signed: -(2(n-1)), -(2(n-1)-1), …
  - Unsigned: …, \(2^{n-2}\), \(2^{n-1}\), 0, 1, …

- Real
  - Fixed-point
  - Scaled
  - Floating-point
### Character
- **ISO 8859-1**
  - 8 bits/character
  - 256 possible characters
  - encodes Latin alphabet
    - e.g. works for French, but not for Russian
  - most widely supported encoding in US
- **Unicode**
  - 8..32 bits/character
  - up to ~4 billion possible characters
  - encodes (potentially) all human language characters
    - (and even some nonhuman ones...)

### String: sequence of characters
- portable, if you also know:
  - order
  - length, from one of:
    - count ("here come N characters")
    - delimiter (end-of-string character)

---

### "Printable" Text
- Subset of possible 1-byte characters
  - no "control" codes: newline, bell, tab, etc.
    - glyph always matches stored value
  - no "unAmerican" characters: Å, ñ, £, etc.
    - glyph read/writable on any I/O device
- Most portable type
"Binary Text"
- Bitwise conversion: bytes ↔ text
- E.g.: 4 bits ↔ hexadecimal character
  - 0000..1001 ↔ 0..9
  - 1010..1111 ↔ A..F

Most portable "byte stream"
- Inflation: byte becomes >1 character
  - Less if larger radix
    - Hex → 2x
    - Base-64 (e.g. uuencode) → 1.25x
- Need printable chars for delimiters

Same bits, different types

<table>
<thead>
<tr>
<th>Binary</th>
<th>Unsigned Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1100000000100100100001111111011011</td>
<td>3,226,013,659</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hexadecimal</th>
<th>Signed Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0490FDB</td>
<td>-1,068,953,637</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ISO Latin-1</th>
<th>IEEE Floating-point</th>
</tr>
</thead>
<tbody>
<tr>
<td>À I control-O Ù</td>
<td>-3.1415927</td>
</tr>
</tbody>
</table>

More information about computer numbering formats:
Do you recognize these file formats?

```
("root" "x" "0" "0" ":/" ":/"
("uucp" "x" "10" "14" "uucp" ":/" ":/"
("fido" "x" "501" "501" ":/" ":/
```

```
"REVIEW_DATE","AUTHOR","ISBN","DISCOUNTED_PRICE"
"1985/01/21","Douglas Adams",0345391802,5.95
"1990/01/12","Douglas Hofstadter",0465026567,9.95
"1999/12/03","Richard Friedman",0060630353,5.95
"2001/09/19","Karen Armstrong",0345384563,9.95
"2002/06/23","David Jones",0198504691,9.95
"2002/06/23","Julian Jaynes",0618057072,12.50
"2003/09/30","Scott Adams",0740721909,4.95
"2004/10/04","Benjamin Radcliff",0804818088,4.95
"2004/10/04","Randel Helms",0879755725,4.50
```

Metaformats (1)

- **DSV** – delimiter-separated values
  - **CSV** – comma-separated values
    - 1 record per line
    - First line: fields = variable names
    - Remaining lines: fields = variable values

DSV format: separated by colon, one record per line (mainly used in Unix)

```
smith:!:100:100:8A-74 (office):/home/smith:/usr/bin/sh
guest:!:200:0:/:/home/guest:/usr/bin/sh
```

CSV format: mainly used in Windows system

```
"REVIEW_DATE","AUTHOR","ISBN","DISCOUNTED_PRICE"
"1985/01/21","Douglas Adams",0345391802,5.95
"1990/01/12","Douglas Hofstadter",0465026567,9.95
```

Metaformats (2)

- **XML**
  - Simple syntax
  - Plain text
  - Nested
  - Semantics
  - Suitable for complex nested or recursive structure

```xml
<booklist>
  <book ISBN="1-558-28592-x" availability="instock">
    <title>XML: A Primer</title>
    <price>24.99</price>
    <author>
      <name>Simon St. Laurent</name>
      <contactinfo>
        <email>simonstl@simonstl.com</email>
        <website>http://www.simonstl.com</website>
      </contactinfo>
    </author>
  </book>
  <book ISBN="0-130-81152-1" availability="instock">
    <title>The Xml Handbook</title>
    <price>44.95</price>
    <author>
      <name>Charles F. Goldfarb</name>
      <name>Paul Prescod</name>
    </author>
  </book>
</booklist>
```

Scientific data types

A few examples
Scientific data types (1)

Hydrology data types

Scientific data types (2)

Geospatial data
Scientific data types (3)

**Bioinformatics data types**


---

**Scientific data types (4)**

- **Chemical data types**
  - Registry number
  - Chemical name
    - One chemical has many different names
    - Chemical names are sometimes misleading
    - Slight difference in spelling of a chemical name can lead to a complete misrepresentation of the chemical substance
  - Modular formula
  - Structural formula
  - Connectivity tables
Summary of scientific data types

- Disciplinary specific
- Associated with data collection methods
  - Computed
  - Captured by instruments or satellites
- Associated with the natural phenomenon, organism, or object being described by the data
- Grouping of a set of attributes

Data formats
Scientific data formats

- Disciplinary specific formats
- Image formats (2-D)
- Image formats (3-D, MESH)
- Matrix formats
- Microarray file formats
- Communication protocols

See handout for examples

What formats are for?

- Archiving
  - Preservation for posterity
- Storage
  - Availability for “arbitrary” access
- Transmission
  - delivery across
    - hardware
    - software
    - administrative
  - system boundaries
- Analysis
  - availability for processing
Format requirements for archiving

- Critical
  - portable
  - *self-describing*
    - Assume neither software nor hardware that wrote data will be available when data are read.

- Important
  - space-efficient
    - Huge archives must fit in finite space
  - write speed
    - Huge archives must be writable in finite time.

Format Requirements for Storage

- Critical
  - subset retrieval
    - the piece you need is different from the piece everybody else needs

- Important
  - space-efficient
    - probably less fast storage than archival storage
  - portable
    - fast storage may be accessed by multiple hardware/software architectures
Format Requirements for Transmission

- **Critical**
  - convertible
    - easy to get data and metadata into and out of
  - portable
    - *readable* anywhere
  - extensible
    - can add types and structures you didn't think of yet

- **Important**
  - single stream for data and metadata
    - wadding everything up together reduces risk of missing critical piece and not knowing it

Format Requirements for Analysis

- **Critical**
  - works with *your* software
    - e.g.: data/metadata serialized for processing pipeline
    - e.g.: relevant data "chunk" fits entirely in memory (FFT, etc.)

- **Important**
  - works with *all* your software
    - minimize time spent converting format