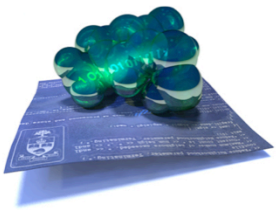
A
Bioinformatics
Course

Abstractions



Boris Steipe

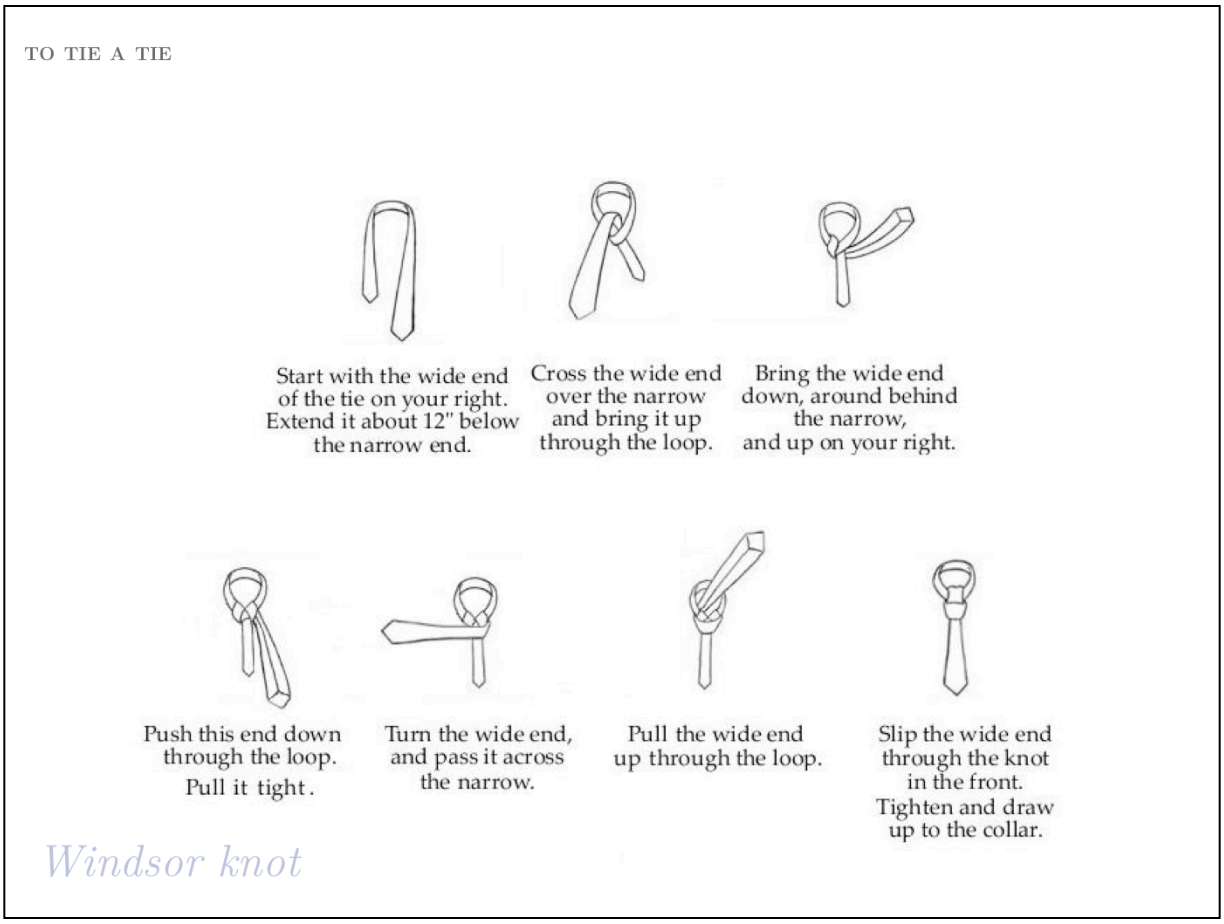
department of biochemistry – department of molecular genetics
University of Toronto

Bioinformatics: Computable Biology

All of Bioinformatics deals with making biology computable, whether we are focussed on the data-management pole, or the computational biology pole of the field.

Making biology computable requires **abstracting biology,** as the first step. This means creating concepts which map to biological entities in a meaningful way, which we can represent in a computer. Such abstarctions can be coarse, or fine grained, they can be powerful, if they represent essential aspects of what we are abstracting in a generalizable, yet compact way, oe they can be ambiguous, lossy, or imprecise. Coming up with good abstractions is not trivial, there are no universal recipes and getting this right requires a fair bit of care. But there is one thing we always must keep in mind: the **abstraction is not the biology**, thus the value of the abstraction always relies on our ability to map it back to a biological entities.

Let's illustrate the power of abstraction with a bit of an idiosyncratic example: how to tie a tie.



Tying a tie is a complicated cultural practice. Passed on by tradition through generations, it involves a sequence of actions that are difficult to describe well and difficult to remember, other than through "muscle memory". This, however, requires frequent practice, something our culture has moved away from. Women wear ties, and men usually only twice in their life – and the last time, the tie is tied for them.

Here is one description of one of the standard knots: the “Windsor Knot”. It goes through seven steps – other descriptions you may find on the Web may have 8 or 9 or even 11 steps, with slightly different emphasis. The narrative description is hard to memorize even though concise but given in free text.

TO TIE A TIE

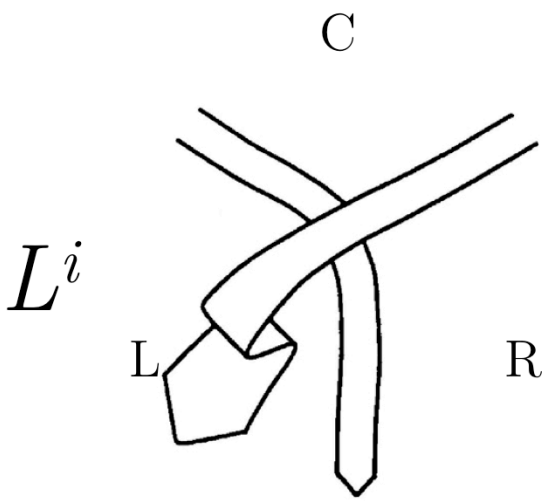
- How do we know we have done it right?
- Is this the best way to tie a tie?
- How do we know how many alternatives there are?
- How can we remember such an *ad hoc* process?
- How can we understand the result?

Can we do better? Not in the sense of a sartorial renaissance of the tie, but better in defining and describing the task at hand?

TO TIE A TIE: ABSTRACTION

What is fundamental and essential ?

- Fixed part and movable part
- Sequence of in-out moves
- Three possible sectors
- Limited number of steps
- Always the same end-move
- Quality: symmetry and balance

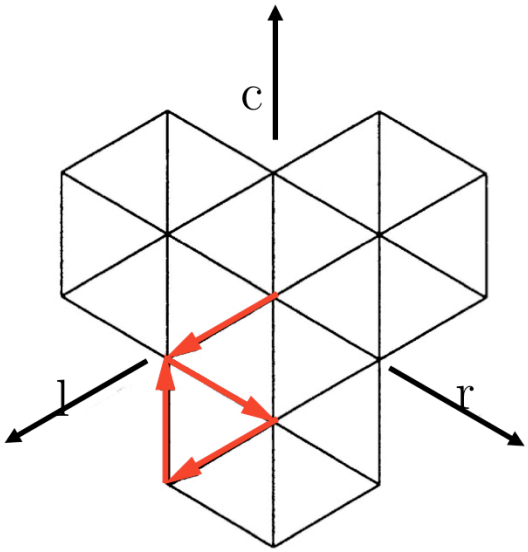


Abstraction:
Persistent, finite, random walk on a triangular lattice !

To define an abstraction, first ask: what is fundamental and essential, and what ca be stripped away.

This was considered by Thomas Fink and Yong Mao, theoretical physicists at Oxford. They modelled tie-tying as persistent, finite, random walks on a triangular lattice.

Persistent, finite, random walk on a triangular lattice.



Measures:

Symmetry:

$$s = \sum_{i=1}^{h} x_i$$

... where $x_i = 1$ for right moves, -1 for left moves. A smaller s is more symmetrical.

Balance:

$$b = (1/2)\sum_{i=2}^{h-1} |\omega_i - \omega_{i-1}|$$

... where $\omega_i(x_i,x_{i+1})$ is the winding direction: $\omega(c,r) = \omega(r,l) = \omega(l,c) = 1$: clockwise move pairs; -1 otherwise. A smaller b is more balanced with fewer reversals of direction.

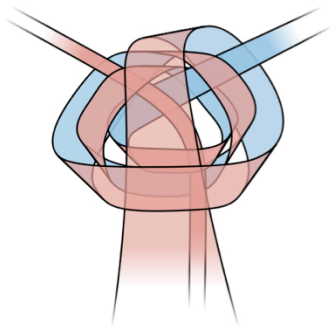
Once the abstraction is defined, it is simple to come up with additional rules ("formulas", or "algorithms") that can be used to compute properties and compare them. Fink and Mao defined measures of *symmetry* and *balance* to be able identify aesthetically acceptable tie knots: good symmetry complements the bilateral symmetry of the face, good balnce prevents the tie from unravelling.

Regarding *symmetry*, the tie is most symmetrical if the number of left moves is equal to the number of right moves.

Regarding *balance*, the tie is well balanced if there are fewer reversals of winding direction.

(Why does the sum have a factor of 1/2 multiplied with it? Is this needed for a physical interpretation of the “balance” measure?)

TO TIE A TIE

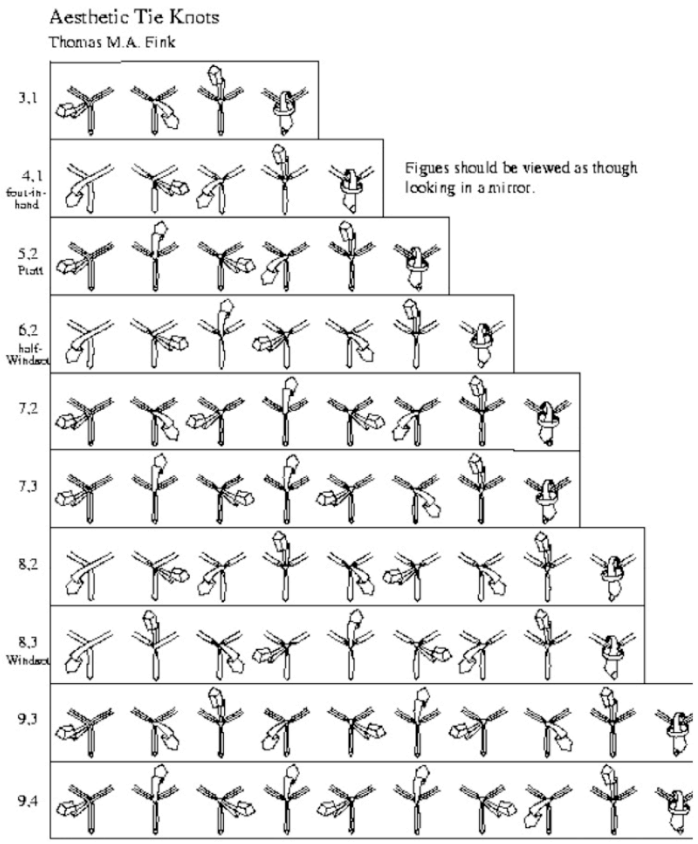


Windsor knot:

$l^i\ c^o\ r^i\ l^o\ c^i\ r^o\ l^i\ c^o\ T$

or

$l^i\ cr\ \ lcr\ \ lcT$



With this theoretical basis (plus a limit on the number of moves, since we are only considering ties of finite length) we can *enumerate* all possibilities and consider the "best".

Ten solutions werere found that can be tied in 9 moves (plus the terminal move) or less and that respect constraints on symmetry and balance. Only four of these had been previously referred to by name in the literature.

Let that sink in: this abstraction and enumeration allowed to expand the universe of tie-knots by 250%, and that in a domain which had been in practical application for decades, if not centuries. And importantly, it allows to draw rigorously complete conclusions: **all** possibilities (that conform to the constraints) have been considered.

(The Internet famous, but utterly abominable "Eldredge knot" is not in this list since it is not tied according to the assumptions of the abstraction. Might as well use a twisty balloon for even greater effect.)

Fink, T.F. & Mao, Y. (1999)
Designing tie knots by random walks. Nature, 398:31-32
http://www.tcm.phy.cam.ac.uk/~tmf20/85ways.shtml

"Fink and Mao have performed a great service for civilization, doing for tie-knot tying what Isaac Newton did for the motion of the heavens: lifting it from the darkness of secrecy, ritual and superstition to the light of rational, scientific good taste."

G. Buck (2000) Nature 403, 362

An enlightened, rational approach ...
... that is what we should aim for in (computational) biology.

ABSTRACTION

"What's in a name? That which we call a rose
By any other name would smell as sweet ..."

In Shakespeare's classic tragedy of romantic love and family allegiance, Juliet encapsulates the play's central struggle in this phrase by claiming that Romeo's family name is an artificial and meaningless convention. Just like with abstractions that describe biomolecules, this is not entirely true: the problems don't derive from the fact that Romeo is **called** a Montague, but that he **is** in fact a member of that family. Even if a *Thing* does not change when its *label* changes, such labels rarely exist in isolation: other *Things* might be referred to by the same label in a category and thus identify relationships. In our example, Romeo would need to renounces his family to acquire a different name.

An even bigger problem – something we encounter frequently in bioinformatics – is the stability of labels: if identifiers are not stable over time, cross-references to that identifier fail. If you decide you'll call a rose a skunk, people would become very confused.

http://en.wikipedia.org/wiki/Romeo_and_Juliet

ABSTRACTION

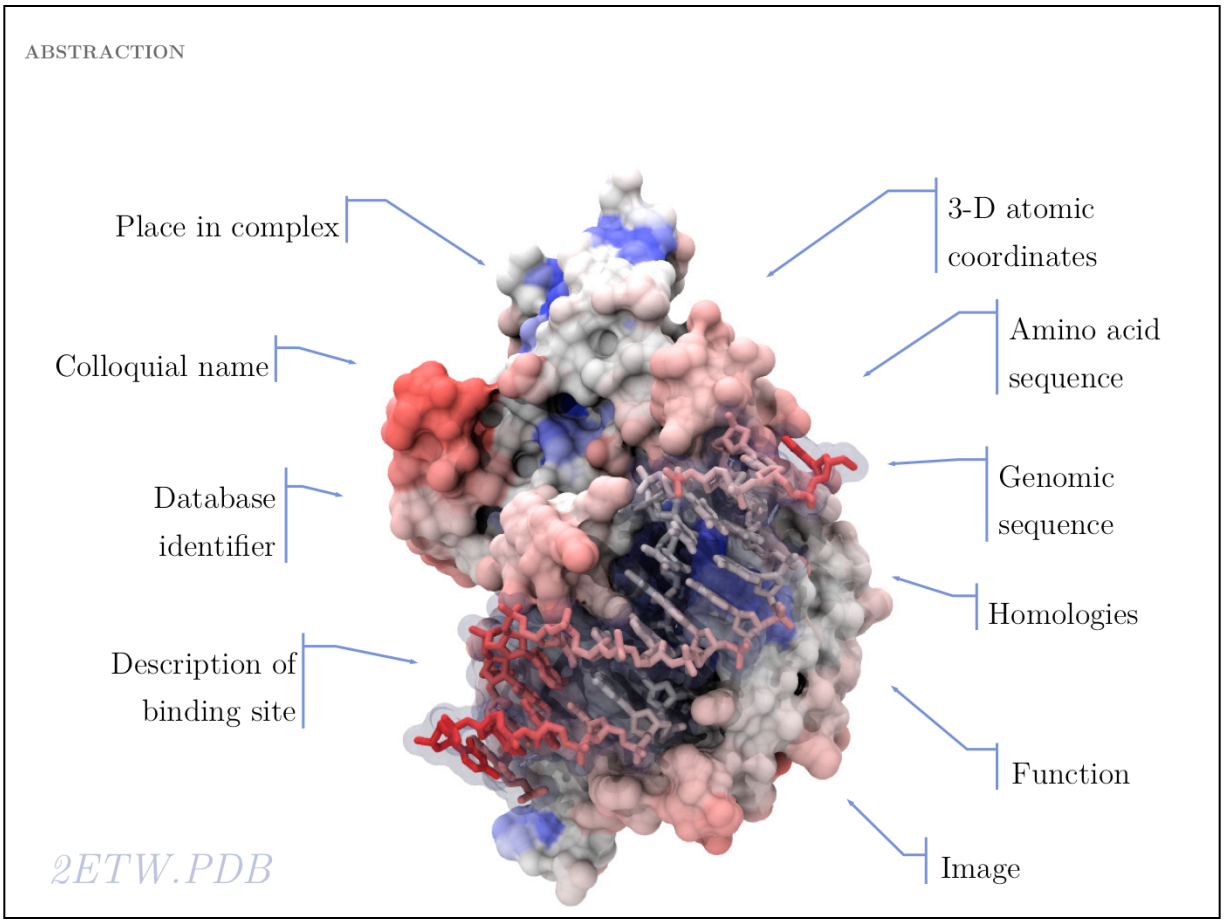
Bioinformatics models biology such that we can compute with its representations.

To compute with such models,

- Representations of biology as data ...
- Semantics of data ...
- Operations with data entities ...
- Metrics of operations ...

... need to be rigorously defined

In order to make biology computable, we have to rigorously define our system of objects and their categories and relationships. This is useful even beyond the requirements of bioinformatics. It is an exercise in clarifying the conceptual foundations of biology itself. In many instances, definitions in current, common use are deficient, either because our current state of knowledge has gone beyond the original ideas we were trying to subsume with a term (e.g. *gene*, or *pathway*), or because an inconsistent formal and colloquial meaning of terms leads to ambiguities (e.g. *function*), or because the technical meaning of terms is poorly understood and generally misused by many (e.g. *homology*).



This image **represents** a particular biomolecule, it was derived from the coordinates of the complex of a yeast sporulation transcription factor bound to its cognate DNA sequence.
What is the best abstraction ?

Asking about *the best* is an ill-posed question if the *purpose* is not specified. There are many possible abstractions, each serving different purposes. Even though abstractions help us model nature by focussing on particular aspects, we must be aware that real molecules have many more properties and features than any single abstraction could capture. Working with abstractions implies we are no longer manipulating the *biological entity*, but its representation. This distinction becomes crucial, when we start computing with representations to infer facts about the original entities. Inferences must be related back to biology! Common problems include (*a*) that the abstraction may not be rich enough to capture the property we are investigating (e.g. one-letter sequence codes cannot represent amino acid modifications or sequence numbers), or (*b*) that the abstraction may be ambiguous (e.g. one protein may have more than one homologue in a related organism, thus the relationship between gene IDs may be ambiguous) or (*c*) that the abstraction may not be unique (e.g. one protein may have more than one function, the same protein name may refer to unrelated proteins in different species).

That said, since the properties of biomolecules derive from their molecular structure, and structure is determined by the sequence of monomer types in the heterocopolymer, **sequence** is the most general abstraction of a gene or protein.

ABSTRACTION

Some examples of abstraction:

... representation of a molecular property
e.g. nucleotide - or amino acid sequence,
3-D coordinates

... description of a function or role
e.g. “transcription factor”,
“checkpoint control element”

... abstract label
e.g. gene name, protein name

Working with abstractions implies we are no longer manipulating the **biological entity**, but its representation.

This distinction becomes crucial, when we start computing with representations to infer facts about the original entities. Inferences must be related back to biology! Common problems include

(a) that the abstraction may not be rich enough to capture the property we are investigating (e.g. one-letter sequence codes cannot represent amino acid modifications or sequence numbers);

(b) that the abstraction may be ambiguous (e.g. one protein may have more than one homologue in a related organism, thus the relationship between gene IDs is ambiguous);

(c) that the abstraction may not be unique (e.g. one protein may have more than one function, the same protein name may refer to unrelated proteins in different species);

(d) that the abstraction is not stable over time and cross-references to an old label may no longer be valid.

COMMON ABSTRACTIONS

Biological Entity	Abstraction	Theoretical domain	Database
Polymer	20 letter amino acid code	String processing	Genbank, GenPept, RefSeq
Molecular conformation	XYZ coordinates, matrices	Floating point processing, linear algebra	PDB
Molecular Interactions	Node-Edge Graphs	Networks, Graph theory	STRING, IntAct
Function	Ontology, DAG	Networks, Graph theory	Gene Ontology
Taxonomy	Hierarchy	Database methods, Graph Theory	NCBI/EBI/DDBJ Taxon
Evolutionary relationship	Tree	Graph Theory, combinatorics	TreeBase

A selection of commonly used abstractions, the domain of computer science they relate to, and common databases that store them.

To structure an abstraction –

we need to define labels; and

we need to structure relationships.

Labels need to refer uniquely to the objects they describe. In practice, natural language is not well suited for that requirement. We need to use "controlled vocabularies" instead. There are two main approaches: numerically controlled vocabularies and synonym constrained vocabularies.

Relationships in Knowledge Engineering are often structured as "ontologies".

"Numerically controlled vocabulary":

A number is defined that uniquely represents the item:

> E.g for a chemical element: the number of protons in its nucleus uniquely defines the element (e.g. calcium→"20");

This approach requires that such unique numbers exist, or can be computed. If that is the case, the numerically controlled vocabulary is usually the ideal solution.

NB. The situation that a unique property of an entity can be concisely described is the ideal case: in that case the identifier captures the most fundamental aspect of the molecule. For calcium, the element does not just **have** the atomic number 20, it **is** the element with 20 protons. Similarly oxytocin does not just **have** the sequence CYIQNCPLG as an attribute, it **is** the peptide with that sequence. However, these are actually favourable exceptions and in general we have to define unique, abstract, and (in principle) arbitrary labels: we call these identifiers.

"Synonym constrained vocabulary":

Use only one form of the string in a database !

> Control the use of labels: e.g. only use "calcium", prohibit the other forms (such as "Calcium", "Ca", "Ca^{++}", "Ca^{2+}" ...);

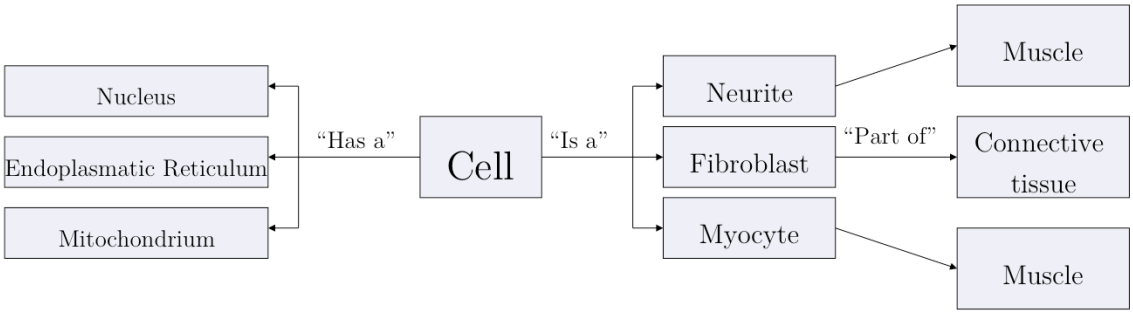
This approach requires that labels are defined, that a mechanism exists to accept instance of these labels and reject others, and that they are known to those who use the abstraction.

Often synonym constrained controlled vocabularies (CVs) are presented as option lists on Web forms. If the CV list is too long for this to be practical, defining the correct form becomes a challenge. In well engineered databases a lot of effort is spent on properly mapping terms to the CV; typically a large dictionary that maps synonyms to the canonical label is employed.

ONTOLOGIES

An ontology is:

- a set of terms (from a **controlled vocabulary**)
- plus a set of **relationships** between them.



Ontologies often represent knowledge bases; they define the semantics of the domain under analysis. This makes ontologies extremely useful for exchanging data between different databases, mapping the "meaning" of a term where its structure can't be mapped.

An ontology is a term used in Knowledge Engineering: a structured CV, with various types of relationships defined between the terms. The "Is a" relationship is probably the most common, but there is really no limit on the types of relationships that may be useful to describe a domain of knowledge. Read more about ...

... the Gene Ontology project: http://www.geneontology.org/

... Open Biology Ontologies: http://obofoundry.org/.

http://steipe.biochemistry.utoronto.ca/abc

B O R I S . S T E I P E @ U T O R O N T O . C A

DEPARTMENT OF BIOCHEMISTRY & DEPARTMENT OF MOLECULAR GENETICS
UNIVERSITY OF TORONTO, CANADA