Two facets of Computer Science: A Personal Experience with the Compute Cycle

Cem Bozşahin
METU Cognitive Science

April 4, 2021

- Horsman et al. (2013) suggested a Compute Cycle

1. First view of the cycle establishes what Computer Programmers do in fixing a program for studying a problem.
2. Second view of the cycle assumes that first cycle is possible, and turns to nature to explore natural problems, keeping in mind what makes first cycle physically possible.

- This has been the implicit understanding of Computer Science for many, for example Turing (1950); Simon (1969); Newell (1973); Marr (1977); Knuth (1996); McDermott (2001); Cockshott et al. (2012)
- In Bozşahin (2016, 2018), I gave it a try in language and planning, implicitly using both cycles.

- Language syntax and human planning abilities seem disparate from an external view.
- Their internal mechanisms appear to be same at some level of abstraction.
- That abstraction, automaton, seems to be relevant to explain planning in humans and chimpanzees, in addition to constraints on language.
- Is recursion the key distinction within species *Homo*? probably not
- More objections: Lobina (2011, 2014); Lobina and García-Albea (2009), among others.

What is at stake in studying recursion and mind?

- Hauser et al. (2002); Fitch et al. (2005); Fitch (2014)
  "Dendrophilia hypothesis"
  Syntactic recursion is the most unique human capacity.
- A one-time cogsci conference was dedicated to recursion in humans:

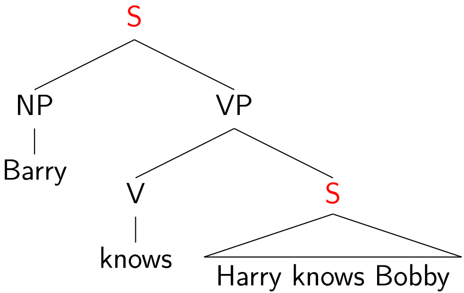
  Speas and Roeper (2009)
- TAG, CCG, LFG, HPSG all use some conception of recursion in theorizing
- Generativists conceive it as operation of the mind.
- Recursion in the lexicon: limits to be explained.

- TAG: factorization of adjunction and substitution.
- CCG: closure of combinations (no fixpoint combinators).
- HPSG: re-entrant unification.
- LFG: Modelling recursion at the functional level.
- MP: move/merge as mind functions?

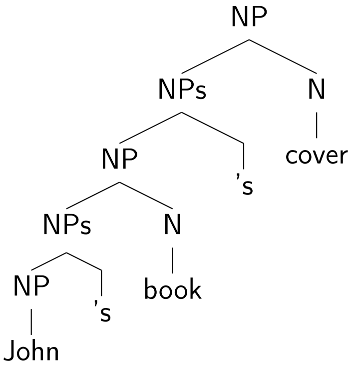
An argument I will not follow

- Lobina and García-Albea (2009): merge is closure.

  Only internal merge (move) is recursive.
- CCG-TAG-G/HPSG: Combinatory capacity does not need movement.
- MP has other things to worry about besides recursion (Kobele and Michaelis, 2009; Stabler, 2010).
- Epstein et al. (1998): move can be eliminated from MP.
- Therefore recursion is probably not necessary.
- Suggestion: what kind of recursion in data can rise from a narrow mechanism?

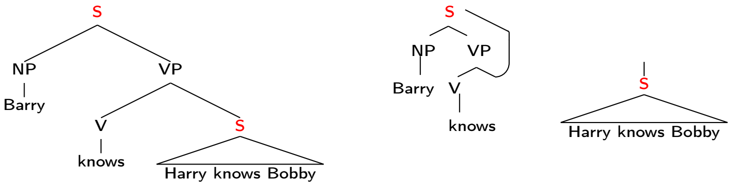


Barry knows Harry knows Jerry knows Bobby.


NP
NPs
N
cover
NP
's
NPs
N
book
NP
's
John

- All these examples are recursion by value
- Another instance of a predicate is taken as a value.
- know, think as lexical items do not refer back to their own structure,

  but to another structure which they take as argument.
- Recursive lexical elements would not be finitely typeable.



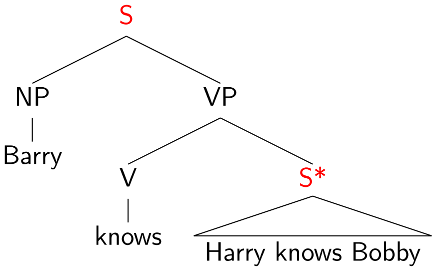
One predication of know on the right,

rather than two, as in recursion by value.

Barry knows Barry knows Barry knows Harry knows Bobby.

Not the native speakers' understanding of knowers and knowees.

Some theories take care of the empirical problem with a single tree:



Two distinct S nodes. (LTAG; Joshi and Schabes, 1992)

Two distinct combination: substitution and adjunction.

The problem is not trees, but not making the distinction.

Some definitions with or without recursion by name.

- Tree: (i) a node called *root* is a tree, denoted as T(root). (ii) The subtrees of a tree T, T(T_1, $T_2, \cdots T_m$), are partitioned into $T_1, T_2, \cdots T_m$, where each T_i is a tree.

- Tree: Any tree is a collection of *nested sets*. A collection of non-empty sets is nested if, given any pair X, Y of the sets, either $X \subseteq Y$ or $X \supseteq Y$ or X and Y are disjoint.
  Knuth (1968: 314)

- $\mathbf{Y} \stackrel{\text{def}}{=} \lambda h.(\lambda x.h(x\ x))(\lambda x.h(x\ x))$ Curry and Feys (1958)
- $\mathbf{U} \stackrel{\text{def}}{=} (\lambda x \lambda y.y(xxy))(\lambda x \lambda y.y(xxy))$ Turing (1937)

- $\mathbf{Y}h = h\ (\mathbf{Y}h) = h\ (h\ (\mathbf{Y}h)) = \cdots$
- $\mathbf{U}h = h\ (\mathbf{U}h) = h\ (h\ (\mathbf{U}h)) = \cdots$

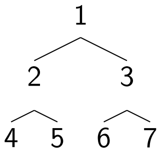
- CCG never employed fixpoint combinators.
- **BST**-system is undecidable if left unconstrained.
- Steedman's principles and constraints (e.g. morpholexical type-raising) provide a near-context-free system.
- Recursion is not constitutive in CCG (unlike TAG).

- $fib(n) = fib(n-1) + fib(n-2)$ $\qquad fib(0) = 0, fib(1) = 1$
- Let
  $fib = \lambda n.\text{if } (n == 0)\ 0 \text{ else if } (n == 1)\ 1 \text{ else } fib(n-1) + fib(n-2)$
- Let $h = \lambda f.\lambda n.\text{if } (n == 0)\ 0 \text{ else if } (n == 1)\ 1 \text{ else } f(n-1) + f(n-2)$
  Then $fib = h\ fib$ because $fib\ n = h\ fib\ n,\ \forall n \geq 0$
  And $fib = \mathbf{Y} h$ because $fib\ n = \mathbf{Y} h\ n, \forall n$, and $\mathbf{Y} x = x(\mathbf{Y} x), \forall x$

Remember how we can eliminate named recursion from non tail-recursive programs:

First eliminate tail recursion

Pre-order Tree Traversal:
1-2-4-5-3-6-7




```
procedure traverse(t:link)
     begin
         if t <> nil then begin
             visit(t);
             traverse(t↑.l);
             traverse(t↑.r)
         end

     end
```

```
procedure traverse(t:link)
     label 0,1;
     begin
         0:if t = nil then goto 1;
         visit(t);
         traverse(t↑.l);
         t:=t↑.r; goto 0

     1:end
```

Then add a stack for `t` and `address`:

```
procedure traverse(t:link)
    label 0,1;
    begin
        0:if t = nil then goto 1;
        visit(t);
        traverse(t↑.l);
        t:=t↑.r; goto 0

    1:end
```

```
procedure traverse(t:link)
    label 0,1,2,3;
    begin
        0:if t = nil then goto 1;
        visit(t);
        push(t); push(3:);
        t:=t↑.l; goto 0;
        3:  t:=t↑.r; goto 0
        1:  if stackempty then
            goto 2;
        lbl:= pop(); t:= pop();
        goto lbl;

    2:end
```

Without a stack we cannot do this.

Now we have no recursion, but, the STACK gives you representation of arbitrary complexity at EXECUTION TIME.

- Why unnatural? It is not finitely typeable

1 read: not finitely representable.

2 means: It has to be specified in the form of a program

3 Curry (1934) suggested axiom schemata, not axioms themselves.

1-3 led to Curry-Howard isomorphism between programs, proofs, types and axiom schemata

- All human knowledge we can think of is finitely representable.
- Human words are finitely representable even if they take same argument types. (by value)

  Bozşahin (2012) suggests some impossible lexical structures.
- Human plans are finitely representable too.

Interim summary:

- Syntactic recursion in language is not recursion by name.
- Syntactic recursion in programming is can be recursion by name.
- Recursion not always constitutive in theoretical linguistics.
- Everett (2005, 2009): Is he really claiming Pirahã people cannot entertain recursive thoughts?

- Recursion by value is limited in humans.
- Recursion by value is attested in non-humans.
- Recursion by value is attested in domains other than language.

- Reactive planning (finite-state)
- Instrumental planning (push-down store of plans, PDA)
- Collaborative planning (embedded push-down store, EPDA)

  Lochbaum (1998); Bratman (1992); Petrick and Bacchus (2002); Steedman and Petrick (2007); Grosz and Kraus (1993); Grosz et al. (1999)

- I-intentions (PDA)
- We-intentions (Searle, 1990) (embedded PDA)

- Can multi-agent planning be sum of plans of single agents? or payoff matrices of agents and conventions? Lewis (1969)

- Searle's (1990) scurrying from rain in the park and corps de ballet in the rain.
- The embedded push-downs of counter-attack in football as common goal
    - A: I chase the ball
    - B: I chase the ball
    - C: I keep close to chasers to join forces
    - D: I keep a watch behind my back
    - E: I keep a watch behind D's back
- No one singly "executes" counter-attack!
- It makes a difference to have a simple stack or stack of stacks (cooperation)

- Finite-state plans: whatever can be afforded by finite history and non-embedded behavior.
- We cannot capture a case where separate actions of an agent match step by step,
- or plans that wait for other plans.

  ex: picking up flowers on the return path of laying them on the floor, ensuring same amount has been picked.

- Instrumental plans: limited context dependency
  In animals and humans
- Julian Jaynes (1976): hapless chimpanzee in captivity.
- Plans that contain other plans and other agents.
- Tomasello et al. (2003): Chimpanzees might have a mind
  but cannot embed minds within minds (no recursion?)
- Formalizing the dependencies in terms of grammars and automata.

- Plans within plans? Jaynes (1976: 219)

$$
\begin{array}{lcl}
\text{S} & \rightarrow & \text{FillWater LureKeeper Spit} \\
\text{LureKeeper} & \rightarrow & \text{Coax} \mid \text{Hail} \\
\text{Coax} & \rightarrow & \text{Stalk}\ \ \text{Coax} \mid \text{AskBanana}
\end{array}
$$

- Spit depends on keeper,
- Coax might fail (spit no more part of this plan)

- External view: finite-state dependencies
- Internal view (planner): contextualized dependencies
- Truly context-free recursion in animals? Probably not (Van Heijningen et al., 2009)
- equiv. to having same dependencies in external and internal view.

- Hauser et al. (2002); Fitch et al. (2005) do not deny non-syntactic recursion in humans

  ((((the hole) in the tree) in the glade) by the stream)
- A closer look at non-syntactic recursion shows striking similarity to language.
- ‘Syntax is most unique human capacity’ argument weakens.
- A certain kind of recursion seems uniquely human.
- Steedman (2014) goes even further, to a narrower class of pushdown machines.

(Scurry-in-rain grammar)

$$
\begin{array}{lcl}
S_i & \rightarrow & \alpha_i \mid A_i \\
A_1 & \rightarrow & \text{run, and do } S_1\text{'s work} \\
\vdots & & \\
A_n & \rightarrow & \text{run, and do } S_n\text{'s work}
\end{array}
$$

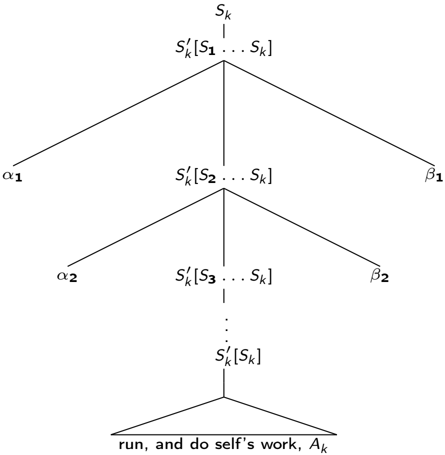
where α_i is a plan with a base case A_i

(Dance-in-rain grammar)

$$
\begin{array}{lcll}
S_i & \rightarrow & S_i'[\pi(S_1, .., S_n)] & \pi(x)\text{: a permutation of } x \\
S_i'[S_j \ldots] & \rightarrow & \alpha_j \; S_i'[\ldots] \; \beta_j & \\
\vdots & & & \\
S_i'[S_i] & \rightarrow & \text{run} & \\
 & & \text{and do } S_i\text{'s work } A_i &
\end{array}
$$

right-hand sides: plans and intentions.
actions and knowledge states are members of RHSs.

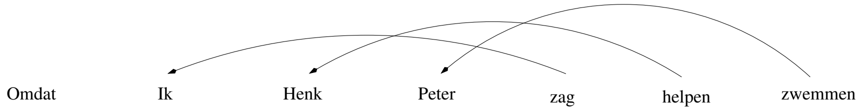
Participant k's grammar for collaborative dancing in the rain.



personal grammar: '$S'_i[S_i]$' rules.
overall collaboration: top rule common to everyone.

α, β contextualize knowledge states and actions (not actions of k)

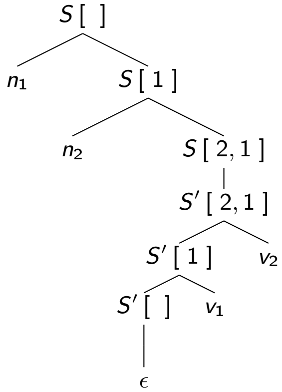
- Such grammars are linear-indexed. (LIG)
- They can handle certain kinds of crossing dependencies.
- And correspond to a stack of stacks as automaton.
- Embedded Push-down Automata Vijay-Shanker (1987); Joshi (1990)



‘because I saw Henk help Peter swim ...’

Dutch and Swiss German style dependencies formalized:

$$
\begin{array}{rcl}
S_{[\ldots]} & \rightarrow & n_i \; S_{[i\ldots]} \\
S_{[\ldots]} & \rightarrow & S'_{[\ldots]} \\
S'_{[i\ldots]} & \rightarrow & S'_{[\ldots]} \; v_i \\
S'_{[\;]} & \rightarrow & \epsilon
\end{array}
$$



$n_1 n_2 v_1 v_2$

- This automata has predictive power:
- $\{a^n b^n c^n \mid n \geq 0\}$ is strictly not context-free but LIG
- no such grammars for $\{www \mid w \in \{a, b, c\}^*\}$,
  or for $\{w \mid w \in \{a, b, c\}^*$ and $|w|_a = |w|_b = |w|_c\}$
  Kanazawa and Salvati (2012),
  $\{a^n b^n c^n d^n e^n \mid n \geq 0\}$ (not a linear metric)

- Instrumental plans and i-intentions: PDA
- Collaborative plans and we-intentions: EPDA ?
- Syntactic capacity in the limit: EPDA (actually linear, LEPDA)

Temporal integration is not found exclusively in language; the coordination of leg movements in insects, the song of birds, the control of trotting and pacing in a gaited horse, the rat running the maze, the architect designing a house, and the carpenter sawing a board present a problem of sequences of action which cannot be explained in terms of successions of external stimuli. Lashley (1951: 113)

- Some internal mechanism seems to be at work

  for planning and language.
- Linear-indexed grammar and associated automata (EPDA) has been the most explicit proposal for what that mechanism might be.
- Not reasoning by analogy: we want to understand limits of natural computation by the mind by explicit proposals for its mechanism from CS.
- Understanding what added explanation can be brought in by a class of automata.
- Such mechanisms seem only depend on recursion by value (natural recursion?)

Conclusion

- Humans appear to be uniquely capable of recursion by value, of the kind that can be afforded by a stack of stacks.
- Various predictions about syntax and other cognitive processes follow from a Computer Science way of thinking about them.
- CS seems to make one more fundamental natural question explicit:

  What evidence do we have that language (or syntax) was there first in the exploits of a potentially common computational substrate?

Bozşahin, C. (2012). *Combinatory Linguistics*. De Gruyter Mouton, Berlin.

Bozşahin, C. (2016). Natural recursion doesn't work that way: Automata in planning and syntax. In Müller, V., editor, *Fundamental issues of AI*. Springer.

Bozşahin, C. (2018). Computers aren't syntax all the way down or content all the way up. *Minds and Machines*, 28(3):543–567.

Bratman, M. E. (1992). Shared cooperative activity. *The Philosophical Review*, 101(2):327–341.

Cockshott, P., Mackenzie, L. M., and Michaelson, G. (2012). *Computation and its Limits*. Oxford University Press.

Curry, H. (1934). Functionality in combinatory logic. *Proceedings of the National Academy of Sciences of the United States of America*, 20:584–590.

Curry, H. B. and Feys, R. (1958). *Combinatory Logic*. North-Holland, Amsterdam.

Epstein, S., Groat, E., Kawashima, R., and Kitahara, H. (1998). *A Derivational Approach to Syntactic Relations*. Oxford University Press, Oxford.

Everett, D. L. (2005). Cultural constraints on grammar and cognition in Pirahã. *Current Anthropology*, 46(4):621–646.

Everett, D. L. (2009). Pirahã culture and grammar: A response to some criticisms. *Language*, 85(2):405–442.

Fitch, T., Hauser, M., and Chomsky, N. (2005). The evolution of the language faculty: Clarifications and implications. *Cognition*, 97:179–210.

Fitch, T. W. (2014). Toward a computational framework for cognitive biology: Unifying approaches from cognitive neuroscience and comparative cognition. *Physics of Life Reviews*, 11(3):329–364.

Grosz, B. and Kraus, S. (1993). Collaborative plans for group activities. In *IJCAI*, volume 93, pages 367–373.

Grosz, B. J., Hunsberger, L., and Kraus, S. (1999). Planning and acting together. *AI magazine*, 20(4):23.

Hauser, M., Chomsky, N., and Fitch, T. (2002). The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298:1569–1579.

Horsman, C., Stepney, S., Wagner, R. C., and Kendon, V. (2013). When does a physical system compute? *Proc. of the Royal Society A*, 470(20140182).

Jaynes, J. (1976). *The Origin of Consciousness in the Breakdown of the Bicameral Mind*. Houghton Mifflin Harcourt, New York.

Joshi, A. (1990). Processing crossed and nested dependencies: An automaton perspective on the psycholinguistic results. *Language and Cognitive Processes*, 5:1–27.

Joshi, A. and Schabes, Y. (1992). Tree-adjoining grammars and lexicalized grammars. In Nivat, M. and Podelski, A., editors, *Definability and Recognizability of Sets of Trees*. Elsevier, Princeton, NJ.

Kanazawa, M. and Salvati, S. (2012). MIX is not a tree-adjoining language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 666–674. Association for Computational Linguistics.

Knuth, D. E. (1968). *Fundamental Algorithms, The Art of Computer Programming Vol. 1*. Addison-Wesley, Reading, MA.

Knuth, D. E. (1996). *Selected papers on computer science*. Cambridge University Press.

Kobele, G. M. and Michaelis, J. (2009). Two type 0-variants of Minimalist Grammars. In Rogers, J., editor, *Proc. of FG-MoL 2005. The 10th conference on Formal Grammar and The 9th Meeting on Mathematics of Language*, pages 81–91, Stanford, CA. CSLI Publications.

Lashley, K. (1951). The problem of serial order in behavior. In Jeffress, L., editor, *Cerebral Mechanisms in Behavior*, pages 112–136. Wiley, New York. reprinted in Saporta (1961).

Lewis, D. (1969). *Convention: A philosophical study*. Harvard University Press.

Lobina, D. J. (2011). Recursion and the competence/performance distinction in AGL tasks. *Language and Cognitive Processes*, 26(10):1563–1586.

Lobina, D. J. (2014). What linguists are talking about when talking about... *Language Sciences*, 45(10):56–70.

Lobina, D. J. and García-Albea, J. E. (2009). Recursion and cognitive science: data structures and mechanisms. In *Proc. of the 31st Annual Conference of the Cognitive Science Society*, pages 1347–1352.

Lochbaum, K. E. (1998). A collaborative planning model of intentional structure. *Computational Linguistics*, 24(4):525–572.

Marr, D. (1977). Artificial intelligence: A personal view. *Artificial Intelligence*, 9:37–48.

McDermott, D. V. (2001). *Mind and mechanism*. MIT Press.

Newell, A. (1973). You can't play twenty questions with nature and win. In Chase, W., editor, *Visual Information Processing*, pages 283–308. Academic Press, New York NY.

Petrick, R. P. and Bacchus, F. (2002). A knowledge-based approach to planning with incomplete information and sensing. In *AIPS*, pages 212–222.

Saporta, S., editor (1961). *Psycholinguistics: A Book of Readings*. Holt Rinehart & Winston, New York.

Searle, J. R. (1990). Collective intentions and actions. In Philip R. Cohen, Jerry L. Morgan, M. E. P., editor, *Intentions in communication*. MIT Press.

Simon, H. (1969). *The sciences of the artificial*. MIT press.

Speas, M. and Roeper, T., editors (2009). *Proceedings of the Conference on Recursion: Structural Complexity in Language and Cognition*. Univ. of Mass, Amherst. forthcoming.

Stabler, E. P. (2010). Computational perspectives on minimalism. In Boeckx, C., editor, *Oxford Handbook of Linguistic Minimalism*, pages 616–641. Oxford University Press, Oxford.

Steedman, M. (2014). Evolutionary basis for human language: Comment on "toward a computational framework for cognitive biology: Unifying approaches from cognitive neuroscience and comparative cognition" by Tecumseh Fitch. *Physics of Life Reviews*, 11(3):382–388.

Steedman, M. and Petrick, R. P. (2007). Planning dialog actions. In *Proceedings of the 8th SIGDIAL Workshop on Discourse and Dialogue (SIGdial 2007)*, pages 265–272.

Tomasello, M., Call, J., and Hare, B. (2003). Chimpanzees understand psychological states—the question is which ones and to what extent. *Trends in Cognitive Sciences*, 7(4):153–156.

Turing, A. M. (1937). Computability and λ-definability. *J. of Symbolic Logic*, 2(4):153–163.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59:433–460.

Van Heijningen, C. A., De Visser, J., Zuidema, W., and Ten Cate, C. (2009). Simple rules can explain discrimination of putative recursive syntactic structures by a songbird species. *Proceedings of the National Academy of Sciences*, 106(48):20538–20543.

Vijay-Shanker, K. (1987). *A Study of Tree Adjoining Grammars*. PhD thesis, University of Pennsylvania.