Great Ideas in Computing

University of Toronto CSC196
Fall 2022

Week 8: October 31 - November 4 (2022)

Week 8 slides

Announcements:

- This week we have Professor Daniel Wigdor will give our final guest presentation on Friday. The class will be on zoom starting at 9:30. Zoom link is available in a quercus annoucement.
- The second assignment is now due Wed, Nov 2, at 8AM. Note the due date was inconsistent between what was on a previous slide and what the Assignment stated (Tuesday, 8AM).
- I have begun Assignment 3. See web page.

Todays agenda

- Any comments on Professor Nisarg Shah's presentation regarding fair allocation.
- New topic: search engines and retrieval by association.

New topic: search engines

I think of search engines as a great idea in the sense of being a "killer application". In addition, one specifc great idea is that search engines enable retrieval by association (i.e., hyperlinks).

In doing so I am mostly talking about search engines as they are mainly used today in terms of searching web documents. That is, search engines that take queries (usually in the form of key words or phrases) and produce a *ranked list* of documents.

We won't consider queries such as " search for a document in which a given photo P appears". Or "find all documents where a 'similar' photo appears".

A very challenging "query" would be to ask "Why was this photo taken?". This requires some NLP and then some basic ability to "understand" the question. But still, can we answer a question like this?

Search engines intro continued

I am going to talk about search engines ignoring the importance (and necessity) of having large pools of fast machines, high speed communication and massive storage.

That is, I am mostly going to talk about search engines in terms of their functionality and the basic computational ideas that make them work (so) well. This is another example (like deep neural nets) of a great idea where greatness depended on new technology. Todays *quality* search engines simply could not exist say using the technology of the 1960s and 70s.

It is also an example where its greatness may also be an inhibitor for thinking about how to "significantly" move beyond the *current norm of key word based search*.

From a functional point of view, while the quality of search has greatly improved, the basic ideas behind key word search remains the same since the late 1990s.

A little search engine history

Search engines are part of the topic of "information retrieval" once the domain of library science. Computerized information retrieval has been an application idea since the start of modern computing.

On the web page there is a link to a prophetic July, 1945 Atlantic article "As We May Think" by Vannevar Bush where he envisions something quite close in many respects to the modern web and hyperlinked documents.

The article begins with the following: "Consider a future device ... in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory."

That is, some kind of semi-automated information retrieval has been thought about for over 75 years.

Some quotes from the Vannevar Bush article

There are a lot of anachronisms (in terms of what the underlyong technology will be, gender roles) in this article but more important there are many insightful ideas about the future of accessing information. Here are some quotes from that article.

"Much needs to occur, however, between the collection of data and observations, the extraction of parallel material from the existing record, and the final insertion of new material into the general body of the common record. For mature thought there is no mechanical substitute. But creative thought and essentially repetitive thought are very different things. For the latter there are, and may be, powerful mechanical aids."
Note: Bush is then clearly drawing his line here between human intelligence (and say creating new knowledge) vs retrieving existing knowledge.

More quotes from Bush's article in the Atlantic

"Our ineptitude in getting at the record is largely caused by the artificiality of systems of indexing. ... The human mind does not work that way. It operates by association. With one item in its grasp, it snaps instantly to the next that is suggested by the association of thoughts, in accordance with some intricate web of trails carried by the cells of the brain."

"Man cannot hope fully to duplicate this mental process artificially, but he certainly ought to be able to learn from it. In minor ways he may even improve; e.g., for his records have relative permanency. The first idea, however, to be drawn from the analogy concerns selection. **Selection by association, rather than indexing**, may yet be mechanized."

"Wholly new forms of encyclopedias will appear, ready made with a mesh of associative trails running through them, ready to be dropped into the **memex** and there amplified." Think now of hyperlinks.

"Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and, to coin one at random, memex will do."

The debate as to the nature of information retrieval

In the 1960's and 70', there was a "debate" (albeit not widely discussed outside of those interested in information retrieval) between those who felt that information retrieval (IR) (i.e. finding documents to satisfy an "information need") was a subfield of AI (and more specifically natural language understanding) verses those who thought it could be best realized by more well established combinatorial, algebraic and statistical ideas. Bush seems to have already settled his views well before this debate is taking place

That is, one constituency felt that we needed to be able to "understand" what a document was saying (and what people were requesting) so as to find relevant documents.

The other constituency felt that the claims of many AI researchers were not at all feasible and that again a more statistical/algebraic/combinatorial approach (devoid of any real "intelligence") would produce better results.

The debate continued

I had a course (1967) in IR from Gerald Salton, who (according to Wikipedia) was "perhaps the leading computer scientist working in the field of information retrieval during his time". His group at Cornell developed the SMART Information Retrieval System".

I am not a great historian but I believe the vector space model (which we will discuss) was his idea. Salton was a proponent of the statistical/algebraic/combinatorial approach. I think that he always felt that AI was over-hyped.

So who the debate?

The debate continued

I had a course (1967) in IR from Gerald Salton, who (according to Wikipedia) was "perhaps the leading computer scientist working in the field of information retrieval during his time". His group at Cornell developed the SMART Information Retrieval System".

I am not a great historian but I believe the vector space model (which we will discuss) was his idea. Salton was a proponent of the statistical/algebraic/combinatorial approach. I think that he always felt that AI was over-hyped.

So who the debate?

As of today, it is clear that the approach of the constituency represented by Salton has turned out to be the basis for the way we currently do search in the internet.

The debate continued

I had a course (1967) in IR from Gerald Salton, who (according to Wikipedia) was "perhaps the leading computer scientist working in the field of information retrieval during his time". His group at Cornell developed the SMART Information Retrieval System".

I am not a great historian but I believe the vector space model (which we will discuss) was his idea. Salton was a proponent of the statistical/algebraic/combinatorial approach. I think that he always felt that AI was over-hyped.

So who the debate?

As of today, it is clear that the approach of the constituency represented by Salton has turned out to be the basis for the way we currently do search in the internet. **However, search engines today do incorporate ML into their retrieval algorithms.**

Key word search

At a very very general level, we can think of current search as the following process:

1. A user converts an "information need" into a query (i.e. a set of key words)
2. The search engine is then an algorithm for the mapping: query $\times$ $\{$collection of documents$\}$ $\rightarrow$ $<$ranked list of "relevant documents"$>$.
3. Upon receiving highly ranked documents, the user may choose to refine the query.
4. This process continues until the user is either satisfied or gives up. How often do you have to refine your queries? How often do you abandon a search?

As we discuss the ideas behind key word search in search engines, it should be noted that there are many specific ideas and engine specfic details that go into making a search engine successful (in terms of the quality, speed, and coverage) and these ideas and details are kept reasonably confidential. Why?

Why the secrecy?

There are two main reasons for not disclosing specific search engine ideas and details:

- Not suprisingly, these ideas are trade secrets that give a company an edge
- Perhaps less obvious, knowing exactly how a compnay does its searches allows one to easily spam documents so as to raise their ranking (and hence lower the quality of the ranking).

So please be advised that what I am discussing is just the high level ideas and not the specifics say being utilized by Google or Microsoft.

It clearly took significant progress in technology (i.e., the speed and memory capilities of large numbers of distributed machines) to make key word search as successful as it is today. Equally important, many significant algorithmic ideas plus extensive and ongoing experience with user requests has been necessary for search engine success.

However, collecting information from user interactions is, of course, an important privacy issue.

The challenge of real time information retrieval

In addition to algorithmic ideas used to improve search quality (i.e., precision, recall), commercial search engines are dealing with enormous collections of sites/documents and must return responses in what appears to be "real time".

Estimates of the size of the web vary. One site (WorldWideWebSize.com) provdes daily reports on the size of the web: That site reported "The Indexed Web contains at least 5.42 billion pages (Sunday, 04 October, 2020)" but as of October 14, 2021, it reports "The Indexed Web contains at least 4.81 billion pages". Did the size really decline?

Precision in a set of documents (for an information need) is defined as the fraction of documents that are relevant. In a ranked list we can say that precision means that the higher the rank of the document, the more likely it is to be relevant.

Recall in a set of documents is defined as the fraction of all relevant documents in the set. In a ranked list of retrieved documents (where there can be many thousands of relevant documents), we want the most relevant documents to occur earliest in the ranked list.

Do we want diversity in the documents retrieved?

We may (or may not) want the highest ranked documents to reflect some desired diversity.

For example, what if I provide the query "What did Donald Trump accomplish as US president"? Do I want just what is reported as his positive accomoplishments? Or do I want just the negative aspects of his presidency? Or do I want a diversity of opinions?

Do we want denials of the Holocaust to be presented in the name of diversity and "balance" as some in the Texas leigslature demand? What is a "legitimate" opinion vs a conspiracy theory devoid of facts? Does Elon Musk and Twitter have a responsibility (or right) to censor what is posted?

Diversity continued

If I ask whether the stock market has recently been rising? Do I want some overall assessment, or do I want reports on different sectors of the market?

Even for a more classical and now perhaps a more mundane example, when I ask for reeent information about "jaguars", do I mean the car, the animal or the NFL football team? I probably only want one of these. When I make my request clear, a search engine should avoid ambiguous meanings.

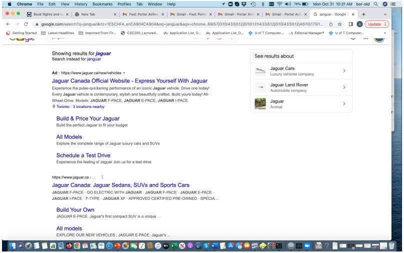


Figure: Search engine reesults for "jaguar" query

Should a search engine use my previous history of requests to better identiffy the most relevant documents personalized for me?

The basic bag of words model

Suppose $\mathcal{C} = \{D_i\}$ is a collection of web documents (URLs).

- We can treat each document as a *bag of words*. Let's just say 200 words per document as some very rough average.
- Each query can also considered as a very small bag or words. Most queries are two or three words. One estimate is an average of 2.2 words per query.
- The most naive approach. Find all the documents that contain all the words in the query. As a naive first approach call these the "relevant documents".
- The most naive way to find all these (potentially) relevant documents would look at each document and check if all the query words occur.
- Even if all the documents were stored locally (which is not possible), what would be a rough estimate for the time to find all the relevant documents?

A quick calculation

You can do a quick calculation: compute $|\mathcal{C}| \cdot \frac{\text{number words}}{\text{document}} \cdot \frac{\text{number words}}{\text{query}}$ and then divide by $\frac{\text{number comparisons}}{\text{second}}$ to estimate the time for naively looking for documents that contain all the query terms.

Let's say that we have approximately 5 billion URLs, 200 words per document, 2 words per query which naively would result in $2 \cdot (10)^{12}$ comparisons. And let's say $(10)^7$ comparisons per second. Then a query would take $2 \cdot (10)^5 = 200,000$ seconds.

OK I might have some miscalculations but clearly this is *not* "real time" and not even feasible.

Making search feasible

One simple idea but but very useful idea is the following. When a search engine *crawls* the web to find documents, it indexes documents so that for each *term* (i.e., word and frequent 2 word and 3 word phrases it maintains a sorted list of documents that contain that term. We usually ignore common articles such as “the”, “an, etc.

A term may also represent a number of strongly related terms. For example, a match for “cook” might be satisfied by “cooking”.

Making search feasible

One simple idea but but very useful idea is the following. When a search engine *crawls* the web to find documents, it indexes documents so that for each *term* (i.e., word and frequent 2 word and 3 word phrases it maintains a sorted list of documents that contain that term. We usually ignore common articles such as “the”, “an, etc.

A term may also represent a number of strongly related terms. For example, a match for “cook” might be satisfied by “cooking”. You can get spelling suggestions, or maybe get a partial match, and sometimes be told that no documents match your query or there are no good matches but still get some suggested matches.

What can happen often is that there be too many documents matching the query terms. So as we already suggested we really need a ranked list of documents in which the “most relevant” documents are ranked highest.

The vector space model and ranking documents

Instead of simply matching for query terms, we want to account for the fact that the occurence of certain terms are more important for relevance.

Gerald Salton's idea was that a document (and a query) are represented by a vector of weighted counts of words/terms. Here are some ways to weight the occurences of terms in a document.

1. Count the number of occurences of a query term in a document, and better yet normalize this count by the relative frequency of terms in "the corpus of documents". This normalized count is called *tf-idf* standing for term frequency-inverse document frequency. Terms that occur infrequently throughout the corpus but appear frequently in a document should be weighted more. Wikipedia quotes a 2015 study that states "83 % of text based recommender systems in digital libraries use tf-idf".
2. Terms that appear in the title of the document or the title of a section heading should be given higher weights.
3. Terms that appear in the *anchor text* are important.

The vector space model continued

The above ideas for weighting terms are independent of the user queries. In contrast, we could also give higher weights to terms that relate to an individuals interests (say as learned by previous searches).

There can be many other ways to weight terms say by using machine learning techniques.

Now once we adopt this vector space representation, we can measure the similarity of a document and a query by say the cosine of these vectors.

An additional idea (in additional to the term similarity of the document and the query) is to expoint the "popularity" of a document. Popularity of a document in Google was done using *page rank* which is basically a random walk on the graph defined by the hyperlinks. This leads to a stationary distribution (i.e., an equilibirum) on the vertices (i.e., the relevant documents).

Some further comments on the history of search engines

Page rank was touted as an essential idea in the early days of Google search but not clear how much of a role it now plays.

At about the same time as page rank, Jon Kleinberg introduced another graph based popularity method called *hubs and auhtorities* which was used in IBMs search engine (which they never commercialized).

With regard to td-idf (now accepted as an important idea), I saw the following comment in a web post (Language Log) https://languagelog.ldc.upenn.edu/nll/?p=27770)
“one of Marvin Minsky’s students once told me that Minsky warned him ‘If you’re counting higher than one, you’re doing it wrong’. Still, Salton’s students (like Mike Lesk and Donna Harman) kept the flame alive.”

Marvin Minsky is recognized as one of the pioneers of artificial intelligence.

End of Monday class and agenda for Wednesday class and a comment on ambiguoous queries

In the Monday class, we mention the old example of the ambiguous query. I tried the query on Monday to see what would result. The query “jaguar” returned mostly links about the car. And, moreover, the top ranked document clearly looks like a sposored ad. Why doesn’t this suprise you?

End of Monday, October 31 class.

Agenda for Wedmesday, November 2 and looking ahead

Agenda for Wednesday, November 2

- Why is search so lucrative; online advertsing.
- The semantic web
- Begin new topic: complexity theory; the $P \neq NP$ conjecture

Looking ahead at the next few weeks

- The $P \neq NP$ issue will lead us to complexity based cryptography.
- Social networks
- A brief selection of some other great ideas.

Why is search so profitable?

Companies such as IBM and (initially) Microsoft did not try to commercialize search, not recognizing the profitability of search. Indeed, should one charge for information or should the business model be based on advertising? Or it possible that search would not be profitable?

We now know that search has turned out to be extremely profitable for companies based on advertising. The main way that Google and other comapnies sell advertising for search has spawned major research in algorithm design and auction theory. We will say more about auctions, game theory and mechanism design.

We can view the process of assigning queries to advertisers (say wanting to display an *ad* as an *online biparitite graph matching problem*).

When a query arrives it needs to be assigned to one (or more, depending on how many advertising slots will be displayed) ads.

The “adwords” assignment problem

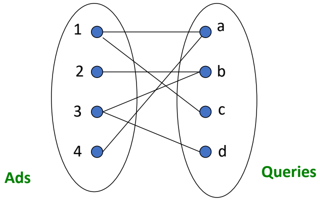


Figure: Figure taken from USC lecture notes by Rafael Ferreira da Silva

Each advertsiser may have a budget (say for a given day) and indicates for given queries (or keywords) what it is willing to pay for that query but never exceeeding its budget for all the queries assigned to that advertiser.

The search engine adjusts this advertiser *bid* for a query based on how well it thinks the ad matches the query and then decides whether or not to assign an advertising slot to an advertiser and the price paid by the advertiser (depending on the slot) for each click by search users for the ad.

The semantic web

We will end our discussion of search engines about where we began when I said, like other great ideas, sometimes these great ideas become so entrenched that it is hard to make further progress.

Is this the case with key word search? What kinds of "indformation needs" are beyond today's search engines? See 2008 "Ontologies and the Semantic Web" article by Ian Horrocks and also his 2005 Lecture by the same title.

The vague goal of the semantic web is "to allow the vast range of web-accessible information and services to be more effectively exploited by both humans and automated tools."

A more specific goal is to *integrate* information that occurs in the web but not in one document.

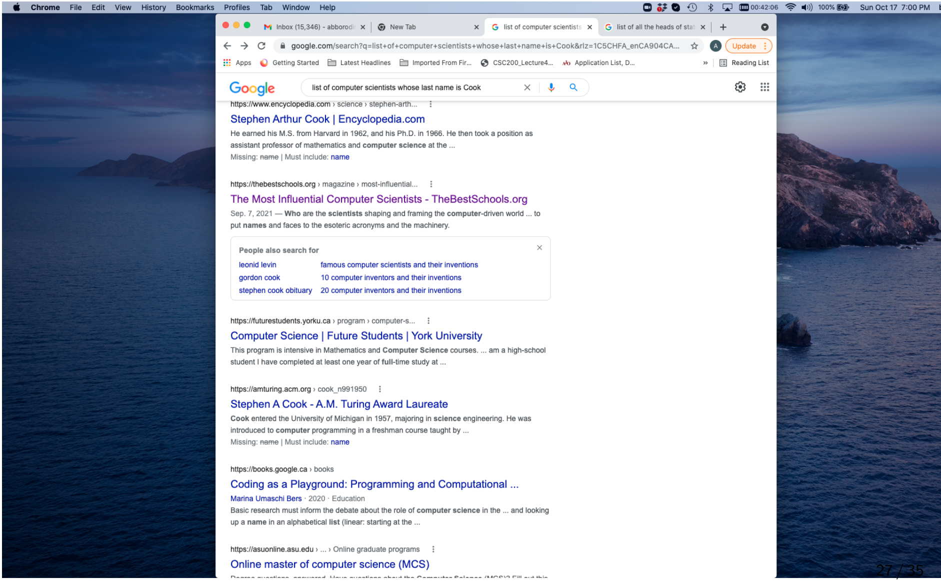
Some specific examples of information that might not exist in any one document

One example Horrrocks gave is to retrieve a "list of all the heads of state of EU countries". Of course, once such an example is given, it is likley (as in this example) that one can successfully find the required information in a single query. (Why was this a difficult search in 2008 and an easy search today? It was the fourth document in my search on October 17,2021.

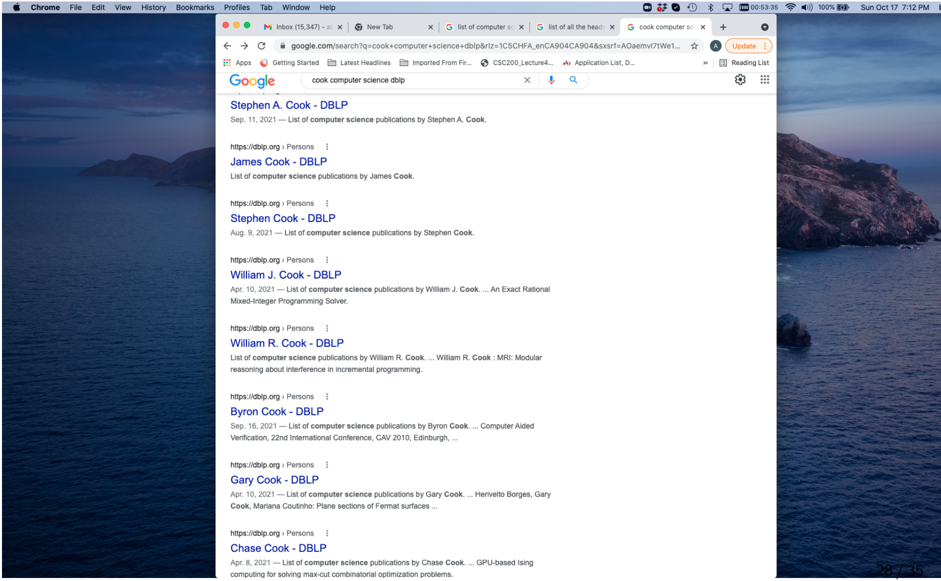
"The classic example of a semantic web application is an automated travel agent that, given various constraints and preferences, would offer the user suitable travel or vacation suggestions". This example still seems beyond something we can easily do with current search engines.

I decided to create the following query "list of all computer scientists whose last name is Cook". In my first search, most of the retrieved documents are not useful but the first of the retrieved documents is for Stephen Cook and the second document is a very incomplete list of computer scientists.

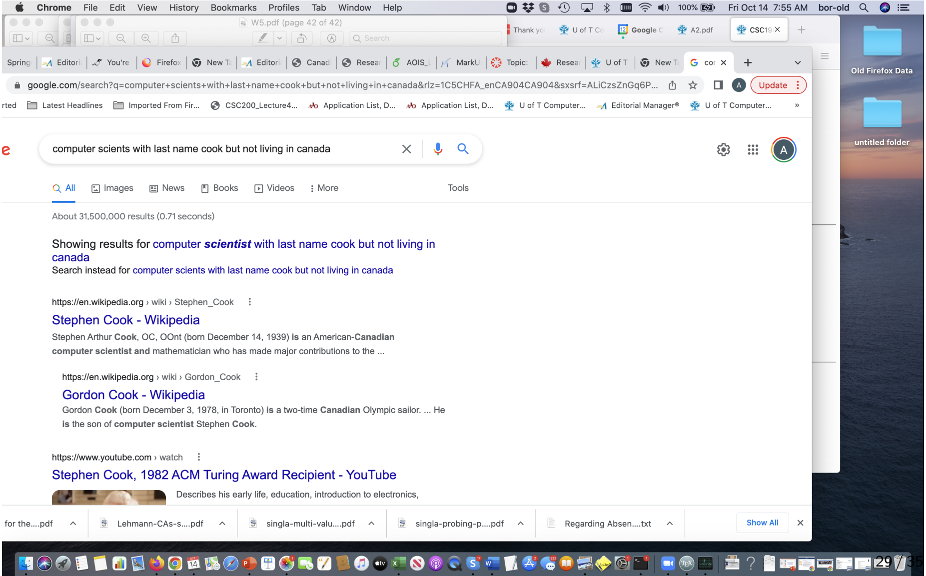
Screenshot of my query for computer scientists with last name Cook



Another search to find other computer scientists with last name Cook



October 14, 2022 search to find a computer scientist named Cook not living in Canada.



The photo query

Last class, we briefly mentioned that today one can have a query such as "find me all documents where this exact photo exists" or "find me a document that contains a photo closest to the query photo".

If the photo comes from a document with text an especially if the photo has a caption, we might already have enough informartion about the photo to do a key woed search.

What if the photo is just something you scanned? One way this can be done (and perhaps this is the main idea) is to treat the photo as a vector compriseds on the pixels. Then we can have a indexed list of photos (each represensented as a vector) and then the problem becomes a well studied problem in computational geometry; namely, the problem of *nearest neighbour search in high dimensions*.

Complexity theory; the extended Church-Turing thesis

We recall the Church-Turing thesis, namely that every computable function f is Turing computable. More precisely, there is a Turing machine M such that on every input x, M halts and outputs $f(x)$. That is, the Church-Turing thesis equates the informal concept of "computable" with the matehmatically precise concept of "Turing machine computable".

The extended Church-Turing thesis equates the informal concept of "efficiently computable" with the mathematical precise concept of computable by a Turing machine in polynomial time".

More precisely, the extended Church-Turing thesis states that a function f is efficiently computable if there is a Turing machine M and a polynomial $p(n)$ such that on every input x, M halts in at most $p(|x|)$ steps and outputs $f(x)$.

Here we are assuming $x \in \Sigma^*$ for some finite alphabet Σ and $|x|$ represents the length of the string x.

The extended Church Turing thesis continued

In what follows, I will use n to be the length of a an input string; $n = |x|$.

Do we believe the extended Church Turing Thesis?

Why we might accept the extended Church Turing thesis

- We can simulate in polynomial time a random access von Neumann random access machine if we say, as we should, that the time for basic operations on bit operands is $O(1)$. This is a robust definition.
- That is, there is a polynomial function $p_2()$ such that if a function f is computable in time $p_1(n)$ on a von Neumann random access machine, then f is computable in polynomial time $p(n) = p_2(p_1(n))$ on a Turing machine. For example, if $p_1(n) = n^3$ and $p_2(n) = n^2$ then $p(n) = n^6$.
- For problems involving say enormous graphs, we may need sublinear time; for other problems we may need linear or near linear times. But as an abstraction, we are saying that a polynomial time algorithm is "efficient".

Why we should be less accepting of the extended Church-Turing thesis

While we are very confidant about the Church-Turing thesis (for defining "computable"), there are various reasons to be a little more skeptical about the extended Chruch-Turing thesis.

- An algorithm running in a polynomial time bound like n^{100} is not an efficient algorithm.
- An algorithm running in an exponential time bound like $(1 + \frac{1}{1000})^n$ is an efficient algorithm for reasonably (but not too) large input lengths. Note: $(1 + \frac{1}{k})^k \to e \approx 2.72$
- While we can simulate classical computers (i.e. von Neumann machines) in polynomial time, we do not know how to simulate non classical computers (e.g., quantum computers) in polynomial time.
- Factoring is an example of a problem that can be computed in polynomial time by a quantum computer whereas we do not believe factoring is polynomial time computable on a classical computer. So it is possible that we will have to change of definition of "efficiently computable" to be polynomial time on a quantum computer.

So should we accept the extended Church-Turing thesis?

We can accept the extended Church-Turing thesis, arguing as follows:

- Polynomial time computable functions usually have reasonably small asymptotic polynomial time bounds; that is, $n, n\log n, n^2, n^3$‘. There are some exceptions (like n^6, but generally speaking we don't usually encounter polynomial time bounds asymptotically bigger than n^3.
- The robustness of polynomial time (in terms of being closed under composition is not sensitive to the precise model of computing and definition of a time step. This enables us to define our concepts in terms of Turing machines (once we restrict outselves to classical computer models). Linear functions are also closed under composition but linear time computation is very model dependent.
- While non-classical models may contradict the thesis, **so far** we do not have non-classical computers (e.g., quantum computers that go beyond a small number of quantum bits) that are practical in a commercial sense.

End of Week 8 slides

We ended on slide 34 having discussed the extended Church Turing-Thesis. In week 9, we will define the classes P and NP of decision problems and discuss the importance of the discussion $P \neq NP$ conjecture.

I have posted the third and last question of Assignment 3.
The question is best understood by looking at the slides for Week 9. I think it is possible to answer the question by looking quickly at the definitions of P, NP and NP completeness, and the stated consequence of assuming $P \neq NP$.

I am available to answer questions (say on Piazza or by email) throughout reading week.

Enjoy reading week. And try to get caught up so that you can do well in the remaining one-third of courses and your exams.