I’m only happy when it rains: why negative results in computer science should be celebrated, not feared!

MSV Incognito guest lecture

Steven Kelk, DKE

Disclaimer

- I am currently in grant-writing mode
- Due to lack of sleep and general stress I am currently in some kind of half-waking dream state
- So if this seems like one big stream of consciousness, that's because it *is*

This is a talk about…

- Computational hardness
- Models
- Star Wars
- Philosophy

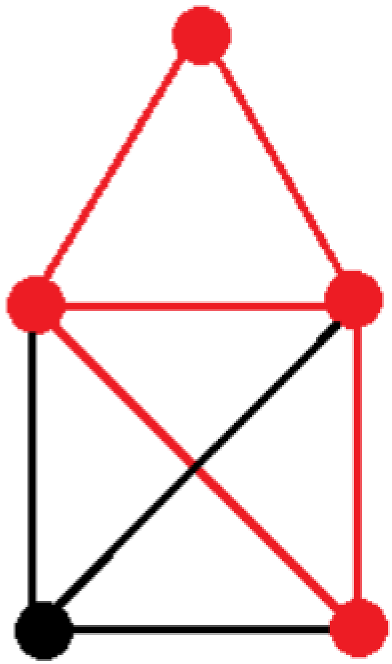
Take-home message

"Only when you understand the limit of what is possible, Can you understand how well, done, you have. Hmmmmmm."

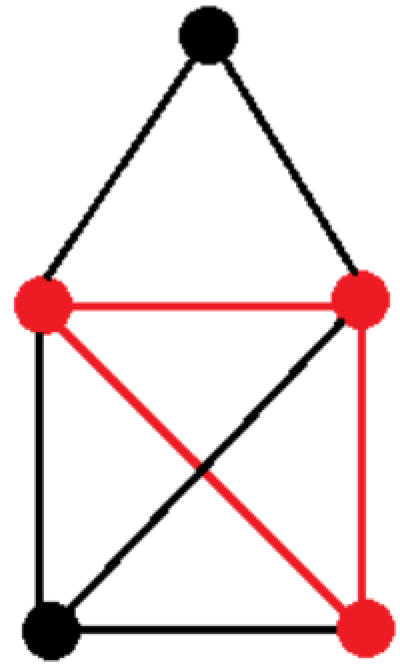
Computational hardness

- In computer science, “hardness” has a specific meaning.
- Example: *maximum clique*

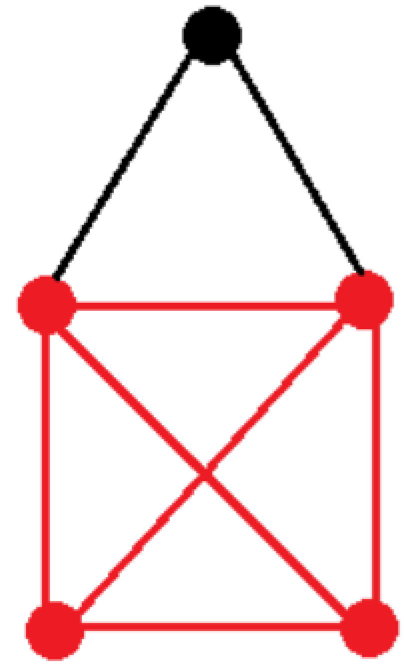
- Input: An undirected graph G on *n* vertices
- Output: The size of the largest *clique* (i.e. complete subgraph) in G



not a clique



non-maximum clique



maximum clique

Computational hardness

- In computer science, “hardness” has a specific meaning.
- Example: *maximum clique*

- There is no polynomial-time (i.e. “fast”) algorithm that can solve *maximum clique*, unless P=NP. (Karp 1972)

Computational hardness

- Can we find an "almost maximum" clique in reasonable time?
- More formally, we say that an algorithm has *approximation ratio c* if, for every input graph G,

$$\frac{Size\ of\ the\ true\ largest\ clique\ in\ the\ graph\ G}{Size\ of\ the\ biggest\ clique\ found\ by\ our\ algorithm} \leq c$$

- A ratio of *n* is trivial to achieve.

J. Håstad, Clique is hard to approximate within $n^{1-\varepsilon}$, *Acta Mathematica 182, 105-142 (1999)*

J. Håstad, Clique is hard to approximate within $n^{1-\varepsilon}$, *Acta Mathematica 182, 105-142 (1999)*

…so in a very formal sense returning a single vertex is best possible ☹


Embrace the darkness…drink deep from
the sweet cup of futility and surrender


(This might just be a personal thing though)

Computational hardness

- Hardness has its roots in the impossibility theorems of Godel and Turing/Church.

- Godel: Incompleteness theorems
- Turing/Church: Negative answer to the *Entscheidungsproblem* (which led to the definition of the Universal Turing Machine – the mathematical foundation of the modern computer)

Computational hardness

- These were clearly “negative results”, but a whole different type to negative results in e.g. statistics:

“We did some stuff. The results were not significant at the p=0.05 level. Shit.”

Computational hardness

- These were clearly "negative results", but a whole different type to negative results in e.g. statistics:

"We did some stuff. The results were not significant at the p=0.05 level. Shit."

Computational hardness

- The next phase of computational hardness emerged in the 1970s: NP-completeness

- This arose as a response to the realisation that some “reasonable” computational problems seemed to require exponential (rather than polynomial) running time, e.g. SATisfiability.

Computational hardness

- If any NP-complete problem can be solved in polynomial time, then all problems in NP (the class of "reasonable problems") can be solved in polynomial time i.e. P=NP
- If we can prove that at least one NP-complete problem requires at least superpolynomial time, then all the NP-complete problems have this property, and P≠ NP

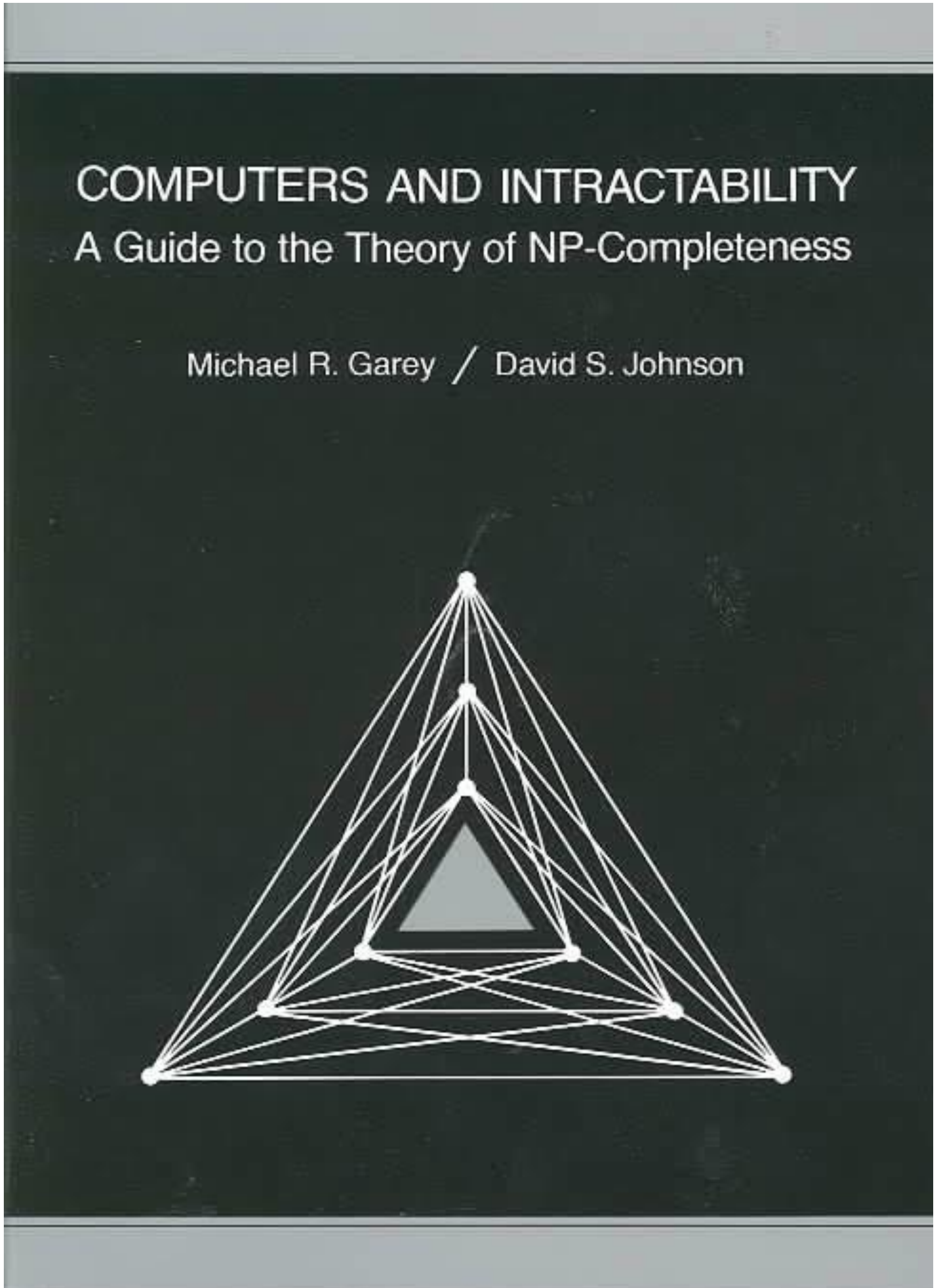
Computational hardness

- There are very many NP-complete problems, and although the majority of scientists believe that they cannot be solved in polynomial time, after 40 years we are not even close to proving this ☹
- In the meantime, proving that a new problem X can “simulate” an NP-complete problem Y, is proof that X is intractable, assuming P≠ NP
- Conditional hardness results

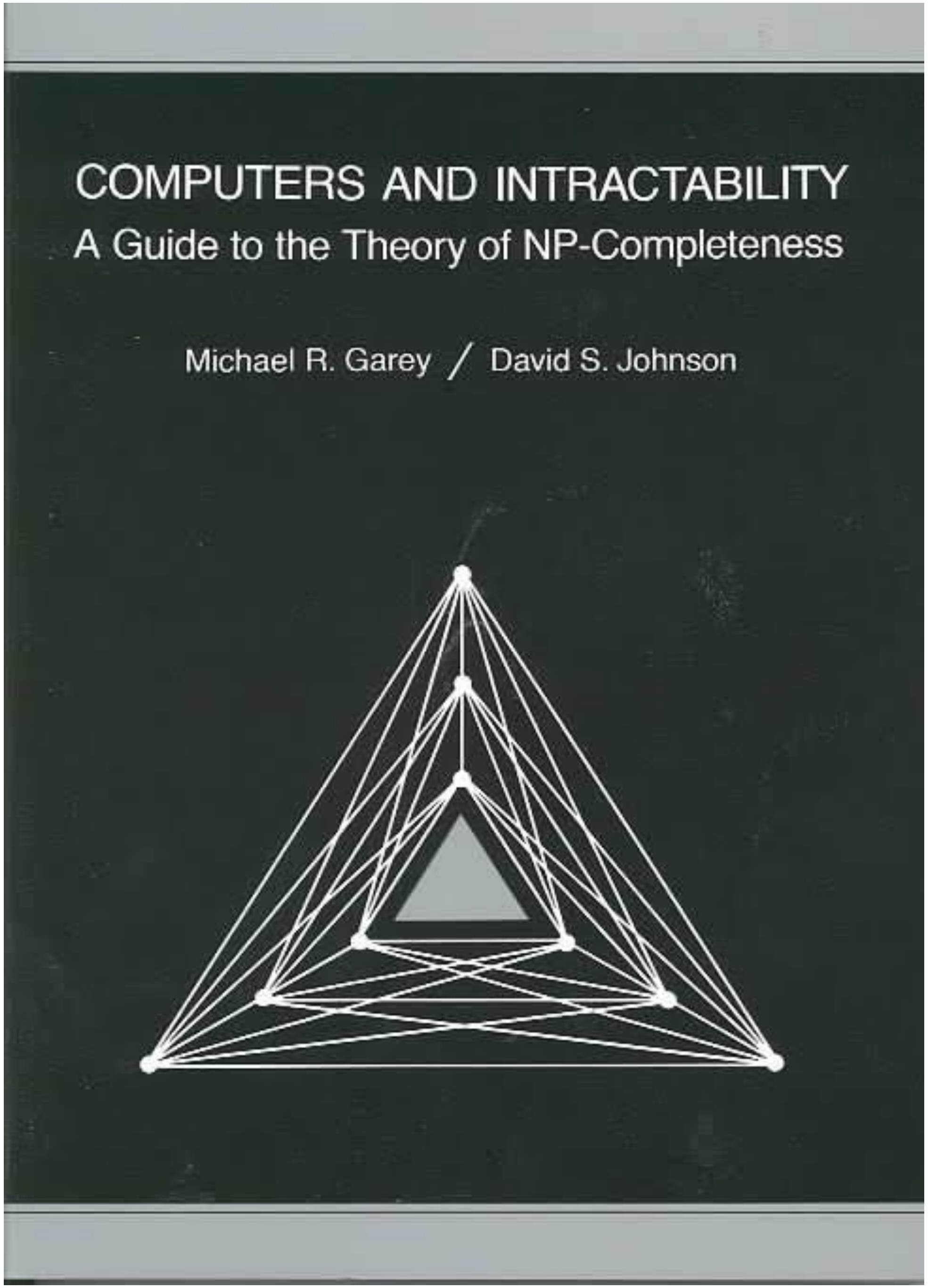
COMPUTERS AND INTRACTABILITY

A Guide to the Theory of NP-Completeness

Michael R. Garey / David S. Johnson



The cover of this book could have been *any* colour.



The cover of this book could have been *any* colour.

But it was *destined* to be black.

Well-known use of hardness results

- “Hey DKE student! I employ you to write fast code, not to look at pictures of cats all day! Why haven’t you come up with a fast exact algorithm yet?”

"I can't find an efficient algorithm, but neither can all these famous people."

(From: Garey and Johnson)


every end
is a new
beginning

Let’s be mindful

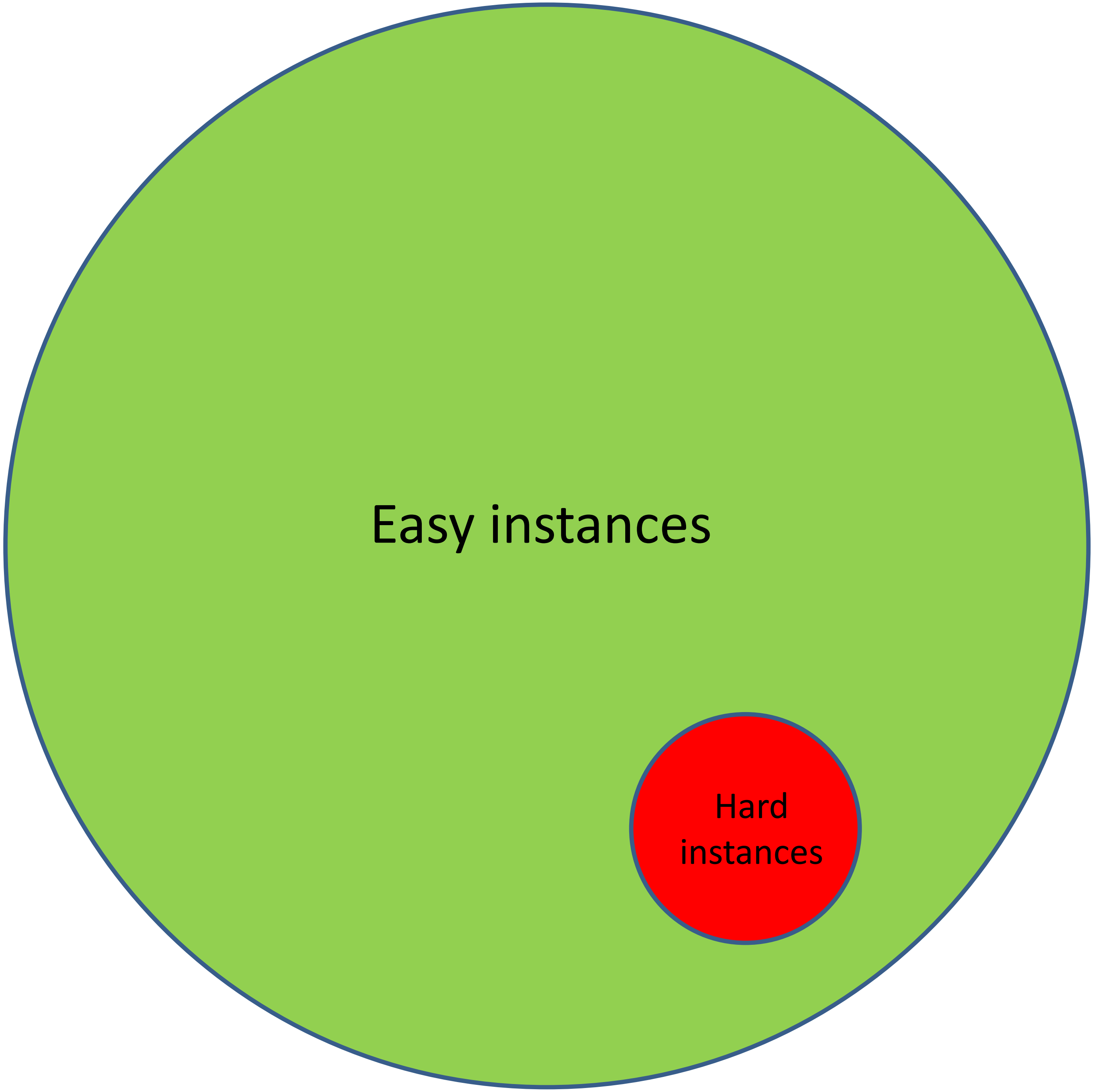
“You, buy happiness, cannot, but you can buy books and, kind of the same thing, that is.”

Hardness is not the end, it's the beginning of an adventure

- "*You probably can't solve NP-hard problems in polynomial time. But in practice you might be able to solve instances of NP-hard problems quite well by exploiting the nature of real-world data and looking for special mathematical structure in your particular instances which let you overcome the hardness, and that's kind of the same thing.*"

Hardness is not the end, it's the beginning of an adventure

- If a problem is hard, then this does not mean "give up"!
- It is an invitation to find out *why* it is hard, and to subsequently design advanced algorithms to tackle it, or (much easier…) avoid it / argue that these nasty cases do not exist in practice


Easy instances
Hard instances

Techniques for tackling hardness

- (Meta-)heuristics ("Hit and hope…")
- Integer Linear Programming (ILP)
- Randomized algorithms
- SAT solvers, Constraint Programming (CP) etc
- Approximation algorithms
- Fixed parameter tractability
- Parallel algorithms
- *Etc.*

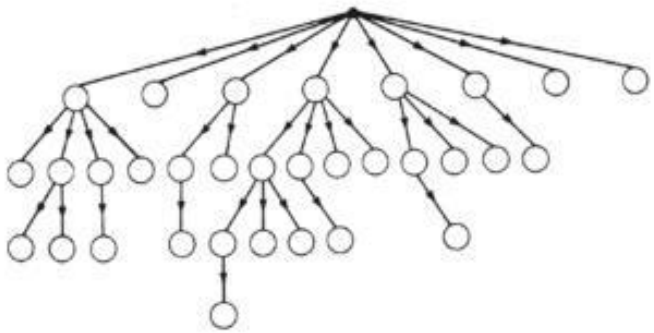
PURE AND APPLIED MATHEMATICS

A Series of Monographs and Textbooks

LINEAR AND INTEGER PROGRAMMING

Theory and Practice

Second Edition



Gerard Sierksma

software included

Applications of SAT Solvers to Cryptanalysis of Hash Functions

Ilya Mironov Lintao Zhang

Microsoft Research
Silicon Valley Campus

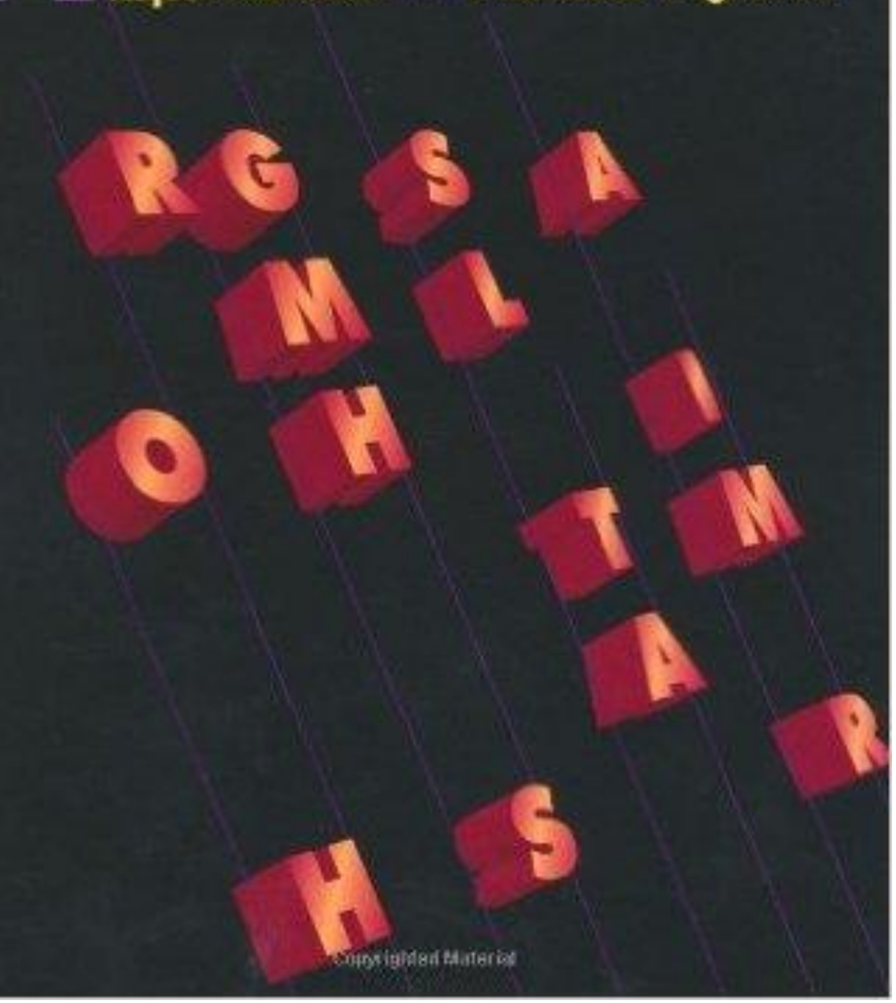
Texts in Computer Science

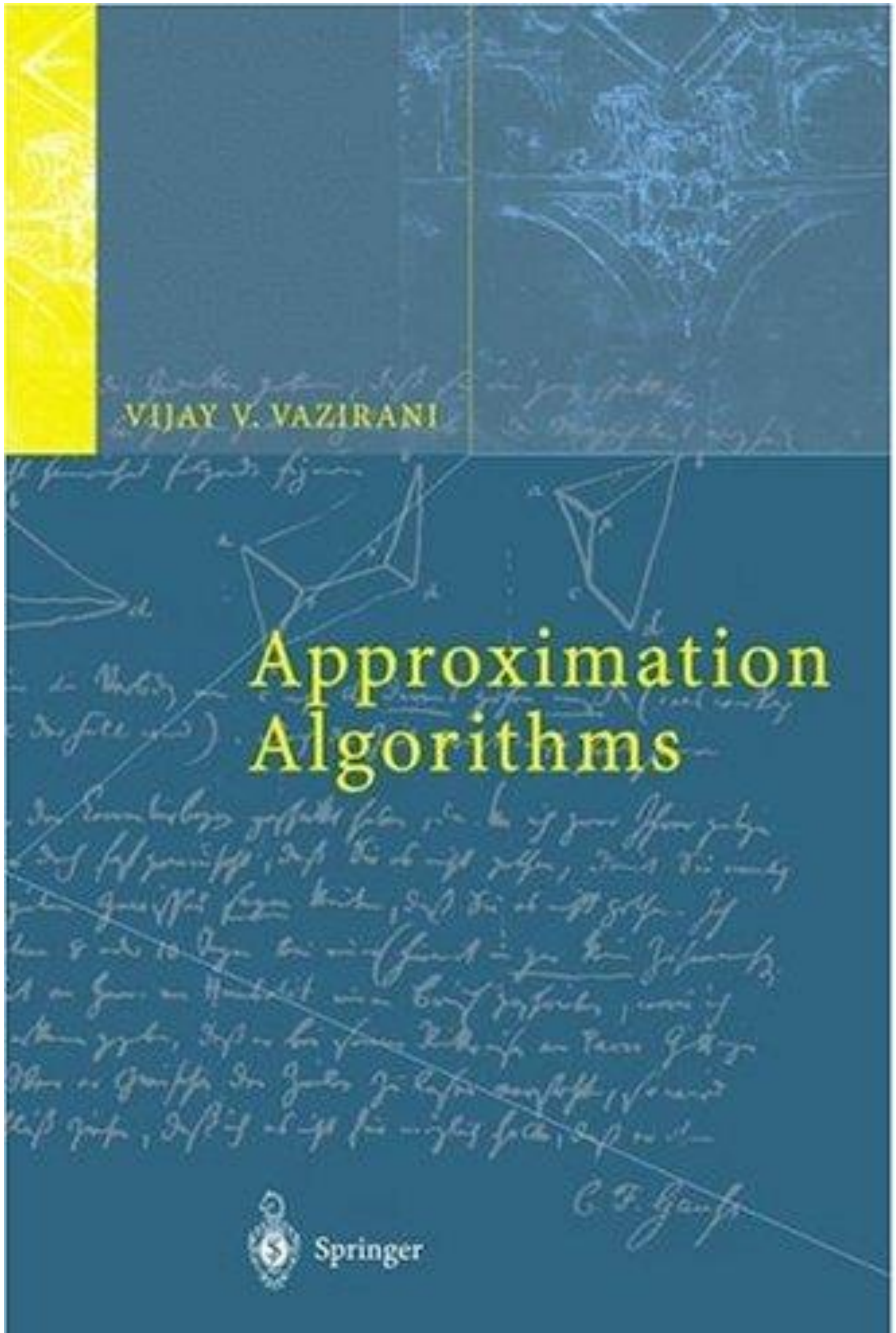
Rodney G. Downey
Michael R. Fellows

Fundamentals of Parameterized Complexity

RANDOMIZED ALGORITHMS

Rajeev Motwani and Prabhakar Raghavan




VIJAY V. VAZIRANI
Approximation
Algorithms
Springer

Techniques for tackling hardness

- (Meta-)heuristics ("Hit and hope…")

- I don't personally develop heuristics, but I understand their use.
  - In many applied contexts, finding a better solution, is already good enough (e.g. reducing costs at a company)
  - This is particularly true when the current solution already "works" and we want to make it "better"

Techniques for tackling hardness

- (Meta-)heuristics ("Hit and hope…")

- I don't personally develop heuristics, but I understand their use.
  - Another case when heuristics are useful is when you are modelling a real-world process and the heuristic "models the phenomenon well in practice"
  - I will come back to this later

Techniques for tackling hardness

- (Meta-)heuristics ("Hit and hope…")

- However, the main problem with heuristics is that you simply have no way of knowing how good your solution is: because we have no model for understanding what optimal solutions look like

Take-home message

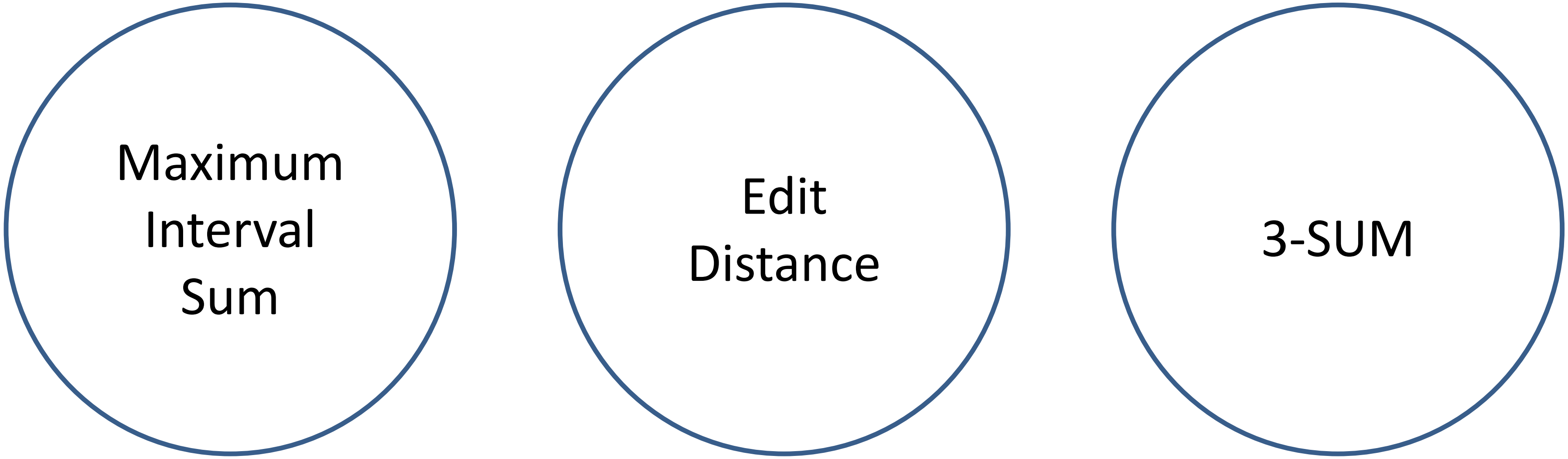
"Only when you understand the limit of what is possible, Can you understand how well, done, you have. Hmmmmmm."

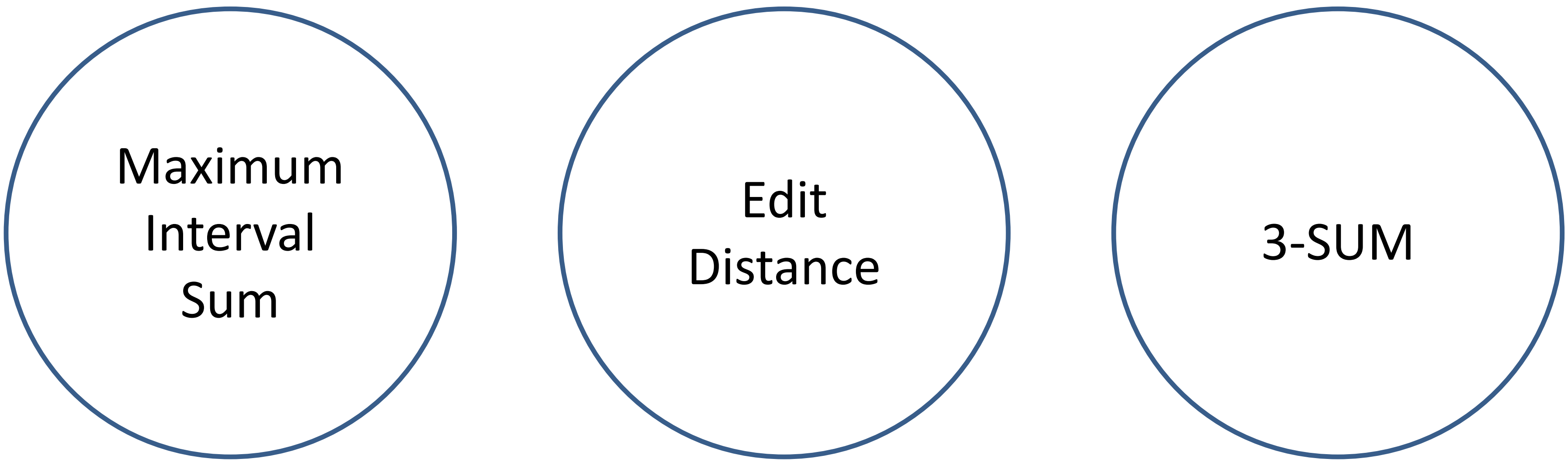
Although they might "work in practice", heuristics do not analyse the limit of what is possible

But Steven, what does this have to do with the real world?

Let’s stop talking about hardness for a moment

- Consider the following three problems. In a formal sense they are “easy” – they all have polynomial-time algorithms.


Maximum Interval Sum
Edit Distance
3-SUM



For each of these problems, it is reasonably straightforward to find an algorithm with running time $O(n^2)$. After all this talk of NP-hardness, that's pretty good right?

Maximum Interval Sum

Edit Distance

3-SUM

It depends what you mean by “good”. If the input data is Big or even Not Small then $O(n^2)$ sucks so bad. And you call yourself a data scientist?

Maximum Interval Sum

Edit Distance

3-SUM

Is it really your fault? Do you suck?

Maximum Interval Sum

Edit Distance

3-SUM

How about this one?

Maximum Interval Sum

Edit Distance

3-SUM

Yes, you suck! By applying dynamic programming $O(n)$ is possible

Maximum Interval Sum

Edit Distance

3-SUM

You missed your chance to solve Big Data!

Maximum Interval Sum: *O(n)* ☺

Edit Distance

3-SUM

How about this one? Can we do better than $O(n^2)$?

Maximum Interval Sum: *$O(n)$* ☺

Edit Distance

3-SUM

We have no idea ☹

Maximum Interval Sum: *O(n)* ☺

Edit Distance

Unsolved problem in computer science:

? *Is there an algorithm to solve the 3SUM problem in time* $O(n^{2-\epsilon})$*, for some* $\epsilon > 0$?

(more unsolved problems in computer science)

We have no idea ☹

Maximum Interval Sum: *O(n)* ☺

Edit Distance

Unsolved problem in computer science:

Is there an algorithm to solve the 3SUM problem in time $O(n^{2-\epsilon})$*, for some* $\epsilon > 0$*?*

(more unsolved problems in computer science)

How about this one? Can we do better than $O(n^2)$?

Image: iStock (edited by MIT News)

Longstanding problem put to rest

Proof that a 40-year-old algorithm is the best possible will come as a relief to computer scientists.

Larry Hardesty | MIT News Office
June 10, 2015

Press Inquiries

Comparing the genomes of different species — or different members of the same species — is the basis of a great deal of modern biology. DNA sequences that are conserved across species are likely to be functionally important, while variations between members of the same species can indicate different susceptibilities to disease.

The basic algorithm for determining how much two sequences of symbols have in common — the "edit distance" between them — is now more than 40 years old. And for more than 40 years, computer science researchers have been trying to improve upon it, without much success.

At the ACM Symposium on Theory of Computing (STOC) next week, MIT researchers will report that, in all likelihood, that's because the algorithm is as good as it gets. If a widely held assumption about computational complexity is correct, then the problem of measuring the difference between two genomes — or texts, or speech samples, or anything else that can be represented as a string of symbols — can't be solved more efficiently.

PRESS MENTIONS

Boston Globe reporter Kevin Hartnett writes that MIT researchers have shown it is impossible to create a faster version of the "edit distance" algorithm, which is used to compare the genomes of different species. Hartnett writes that the finding "has been greeted with something like relief among computer scientists."

The Boston Globe

RELATED

Paper: "Edit distance cannot be computed in strongly subquadratic time (unless SETH is false)"

Backurs and Indyk, Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false), *Proceedings of STOC 2015*

Backurs and Indyk, Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false), *Proceedings of STOC 2015*

SETH = Strong Exponential Time Hypothesis


Revenge of the SETH

Maximum Interval Sum

Edit Distance

3-SUM

All these mad Jedi skills help us understand how hard these problems really are - and thus, whether $O(n^2)$ can be considered “good”

Maximum Interval Sum

Edit Distance

3-SUM

Let’s get mathematical-philosophical

“Mathematical models to approximate reality we make. Implemented as algorithms, mathematical models are. The output of the algorithm model reality, how well does, hmm? Herh herh herh.”

Models models and more models

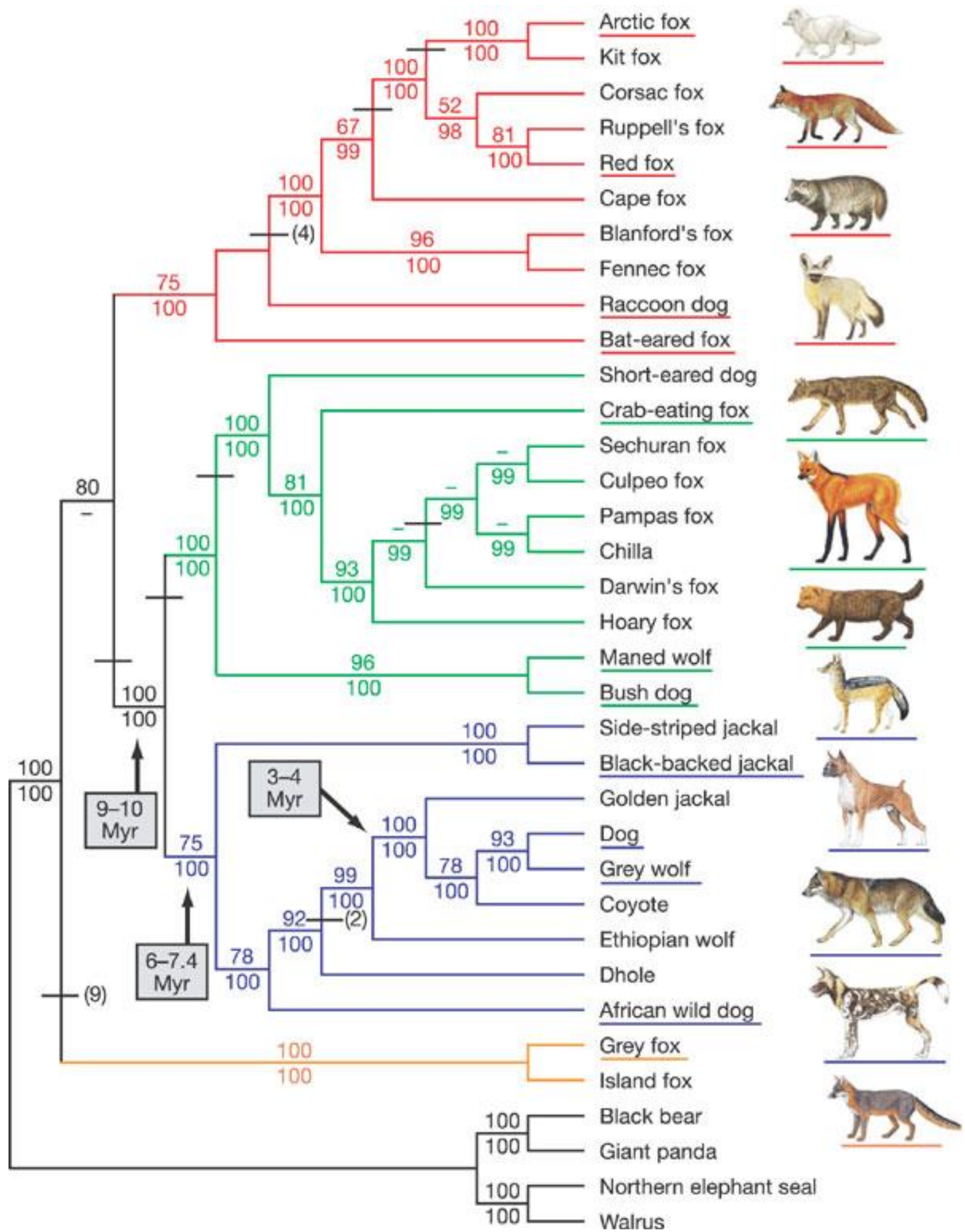
- In many applied sciences, mathematical models are used to approximate reality.
- It is often the case that these models are, implicitly or explicitly, optimization questions.
- This happens if the person who formulated the model, *believes that optimal solutions most accurately reflect reality.*

Models models and more models

- So what does it actually mean if the algorithms that implement these mathematical models, are not finding (or are not guaranteed to find) optimal solutions?
- Many applied scientists who mix computers with mathematical models forget some or all of the following sanity checks:

Sanity checks for models

- Why do I believe that optimal solutions most accurately model reality?
- Does this software package generate optimal solutions? (More fundamentally: what is it trying to optimize?!)
- If not, does it get close? How close?
- If it is not generating optimal solutions, is it the case that “good enough” solutions model reality “well enough”?
- What does “good enough” and “well enough” mean? Am I using circular reasoning?



Genome sequence, comparative analysis and haplotype structure of the domestic dog

Lindblad-Toh et al, Nature 2005

Building evolutionary trees with algorithms

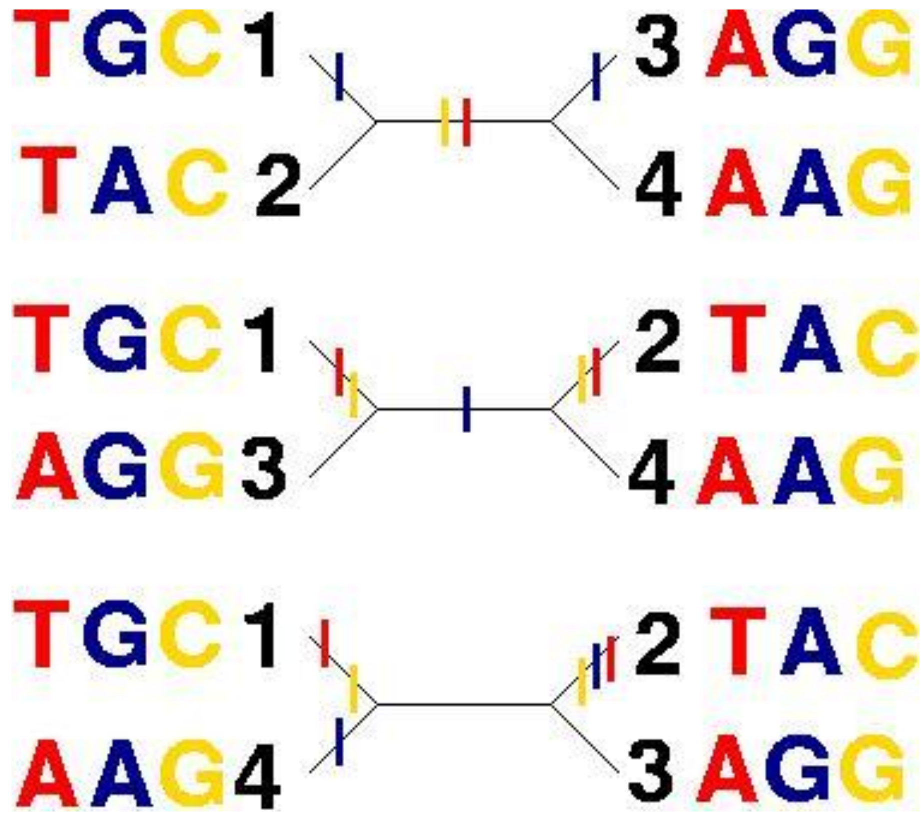
- There are two issues we have to get clear first.
  - What is a “good” tree?
  - Is it computationally tractable to construct such a “good” tree?
- There are many different ways of addressing these questions. The first question in particular is highly subjective.
- There are several major families of tree-building methods, that approach these questions in a different way.

Building evolutionary trees with algorithms

- Maximum Parsimony (ML)
- Maximum Likelihood (MP)
- Bayesian MCMC
- Distance measures

Species			
1 (DOG)	T	G	C
2 (CAT)	T	A	C
3 (FISH)	A	G	G
4 (E.COLI)	A	A	G

Input: a multiple alignment, one DNA string per species



The "most parsimonious" tree solution (4 mutations).
An algorithm that computes optimal solutions to MP, will output this tree.

From: http://artedi.ebc.uu.se/course/X3-2004/Phylogeny/Exercises/mp.html

Building evolutionary trees with algorithms

- Maximum Parsimony (ML) – NP-hard
- Maximum Likelihood (MP) – NP-hard
- Bayesian MCMC - NP-hard (essentially…)
- Distance measures – Polynomial time

Building evolutionary trees with algorithms

- Maximum Parsimony (ML) – Heuristics
- Maximum Likelihood (MP) – Heuristics
- Bayesian MCMC - Heuristics
- Distance measures – Exact algorithms (but not considered a “reliable” method)

Building evolutionary trees with algorithms

- Maximum Parsimony (ML) – Heuristics
- Maximum Likelihood (MP) – Heuristics
- Bayesian MCMC - Heuristics
- Distance measures – Exact algorithms (but not considered a "reliable" method)

- Scientists using these software tools "believe" Bayesian, ML, MP, Distance measures in roughly that order. (MP is no longer cool, *sniff*).

Building evolutionary trees with algorithms

- But if applied scientists prefer to believe the NP-hard models, and all the tools being used are heuristics, what does that actually *mean?*

Building evolutionary trees with algorithms

- But if applied scientists prefer to believe the NP-hard models, and all the tools being used are heuristics, what does that actually *mean?*

- The truth is complex. The trees these software packages output must mean *something,* because there are cases when all the methods recover the same tree, and everyone believes this is the “correct” tree.

Building evolutionary trees with algorithms

- But my worry is that, if you are not careful, you start (unconsciously) defining the “correct tree” as “the same tree that this other famous program produced” – without stopping to ask why we believe the famous program.

Building evolutionary trees with algorithms

- But my worry is that, if you are not careful, you start (unconsciously) defining the “correct tree” as “the same tree that this other famous program produced” – without stopping to ask why we believe the famous program.
- I guess that’s ok, because we all know that aeroplanes can only fly because all the passengers believe that they can ☺

Building evolutionary trees with algorithms

- But my worry is that, if you are not careful, you start (unconsciously) defining the "correct tree" as "the same tree that this other famous program produced" – without stopping to ask why we believe the famous program.
- In practice, "stuff works", but it's still all a bit.....*spooky.*

Finally, for the UCM students!

Finally, for the UCM students!

(They're happy because they do PBL)

Finally, for the UCM students!

- Warning to DKE students: social science alert!
- But at the end I will in completely convincing fashion (ha ha) try to link it back to algorithms.

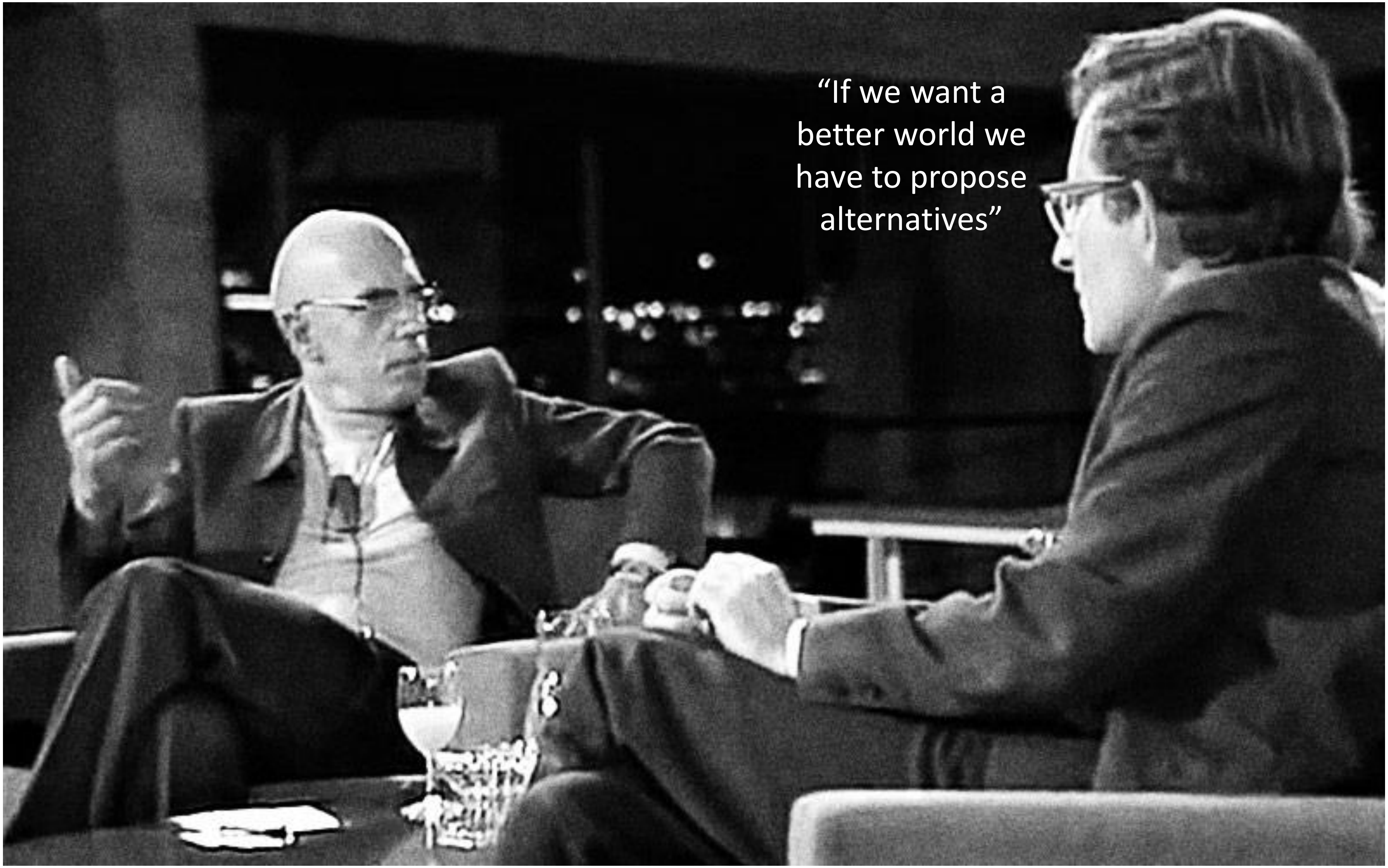
Chomsky vs Foucault

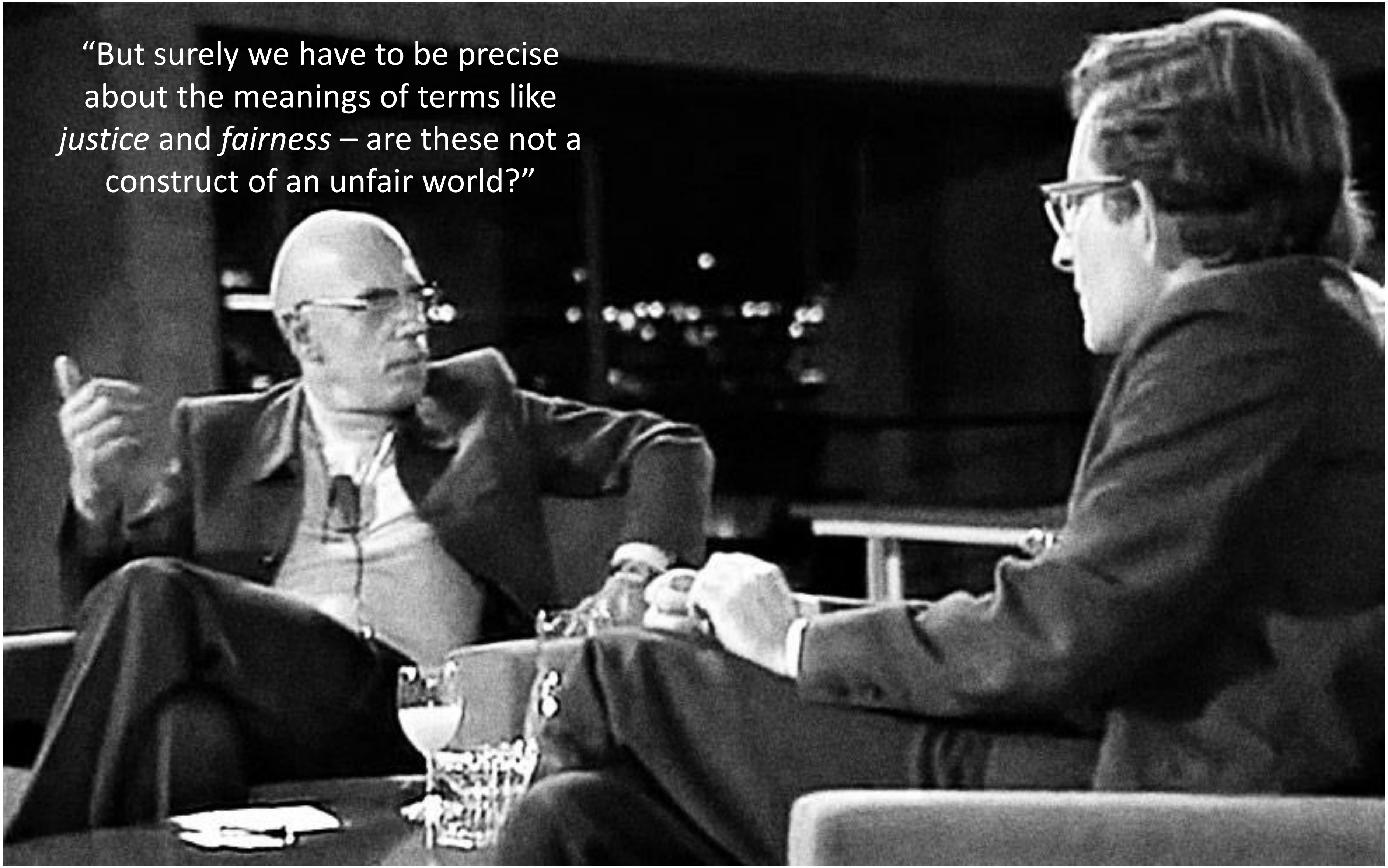
- “Human Nature: Justice versus Power: Noam Chomsky debates with Michel Foucault, 1971.”
- Televised…on YouTube…*very* interesting (even though it does not contain a cat, a robot, a reference to Big Data or a self-driving car).

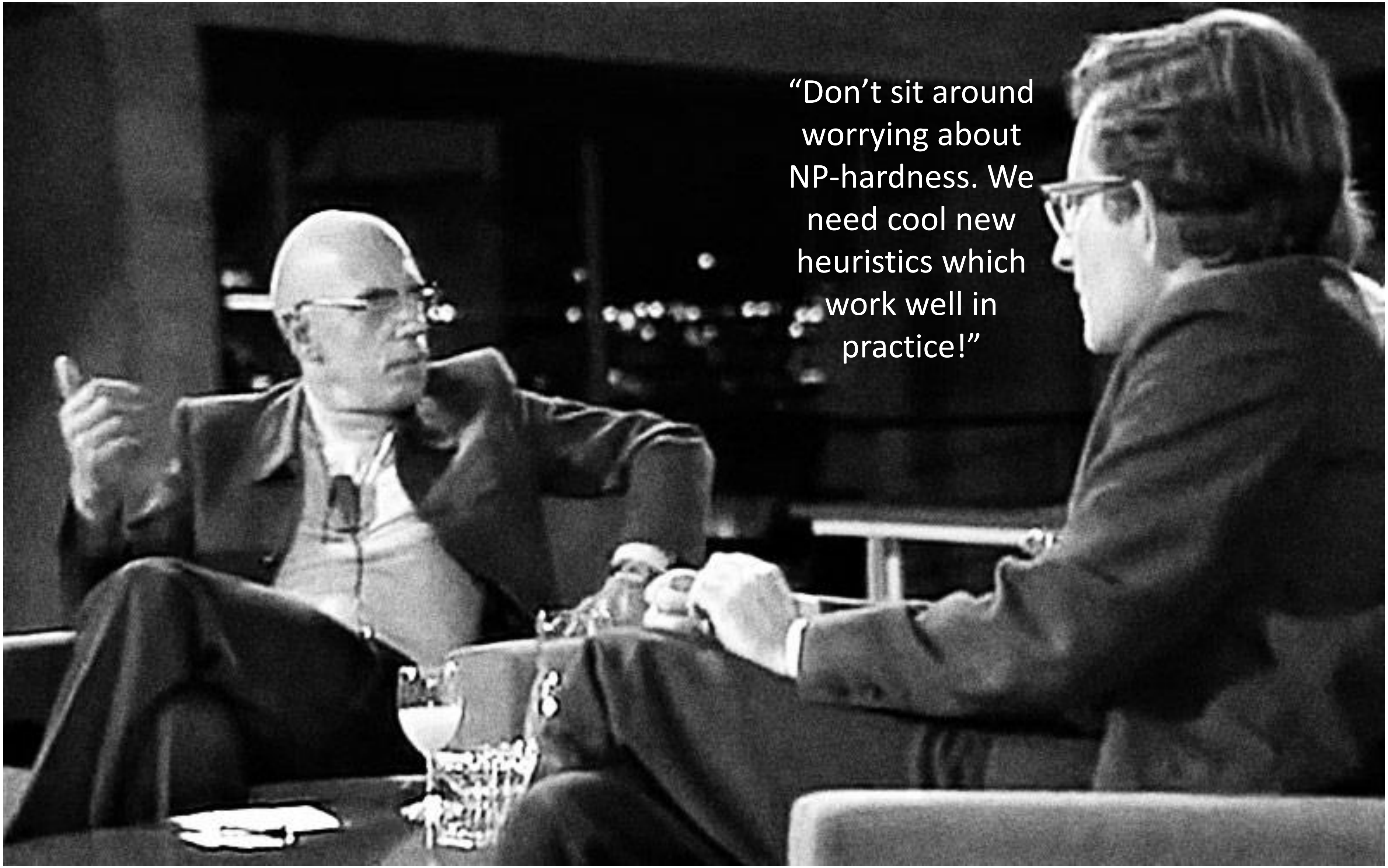
“This, for me, is the key to their fundamental differences. Chomsky is a modernist. Foucault was a postmodernist. The modernist believes ‘justice’ and ‘truth’ have meaning and value independent of power. Human reason, used properly, can lead us toward a more just society. The postmodernist, on the other hand, believes that the meanings of the words ‘truth’ and ‘justice’ are socially constructed largely by those with power to hold and exercise such power. “

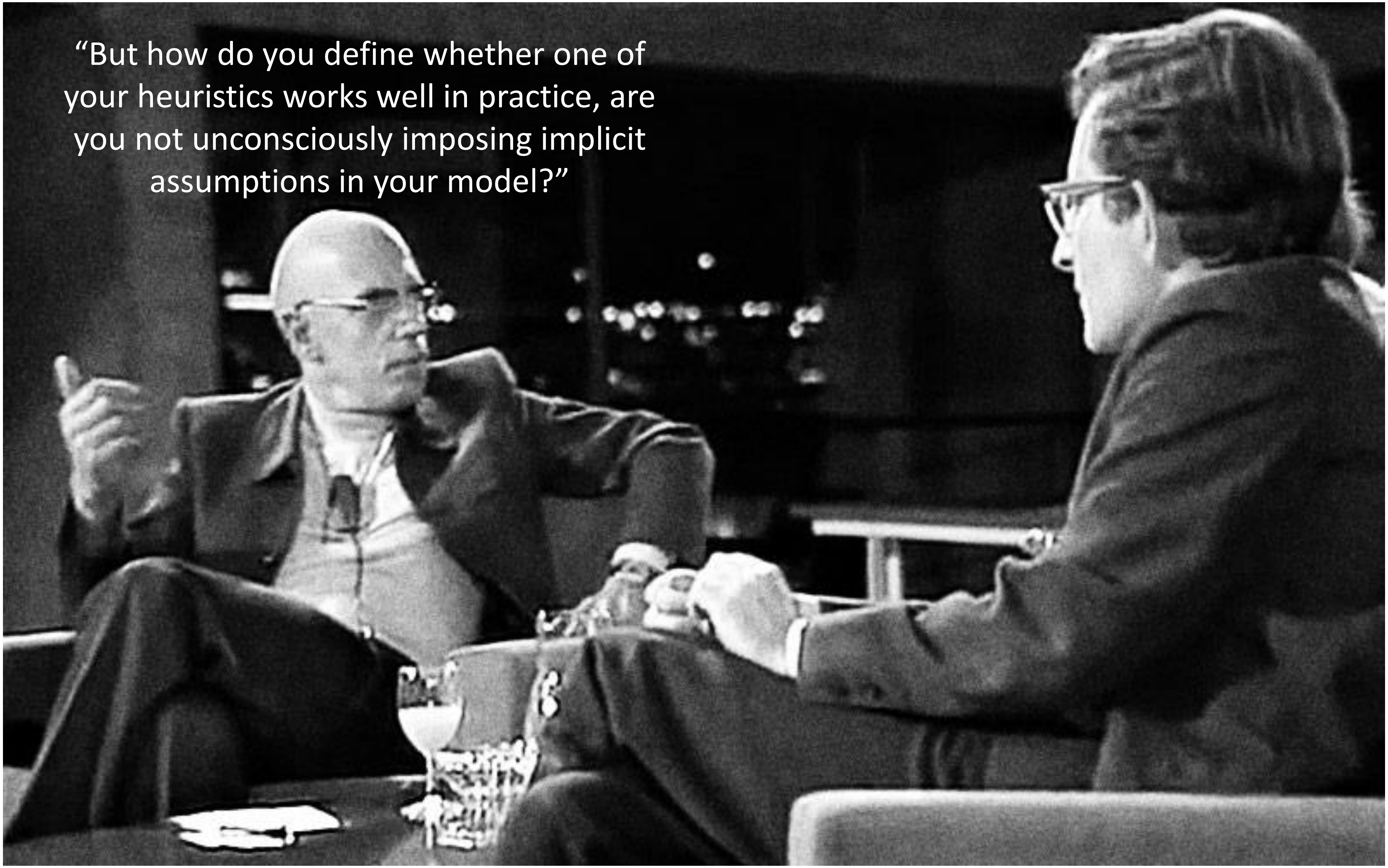
“This, for me, is the key to their fundamental differences. Chomsky is a modernist. Foucault was a postmodernist. The modernist believes ‘justice’ and ‘truth’ have meaning and value independent of power. Human reason, used properly, can lead us toward a more just society. The postmodernist, on the other hand, believes that the meanings of the words ‘truth’ and ‘justice’ are socially constructed largely by those with power to hold and exercise such power. “

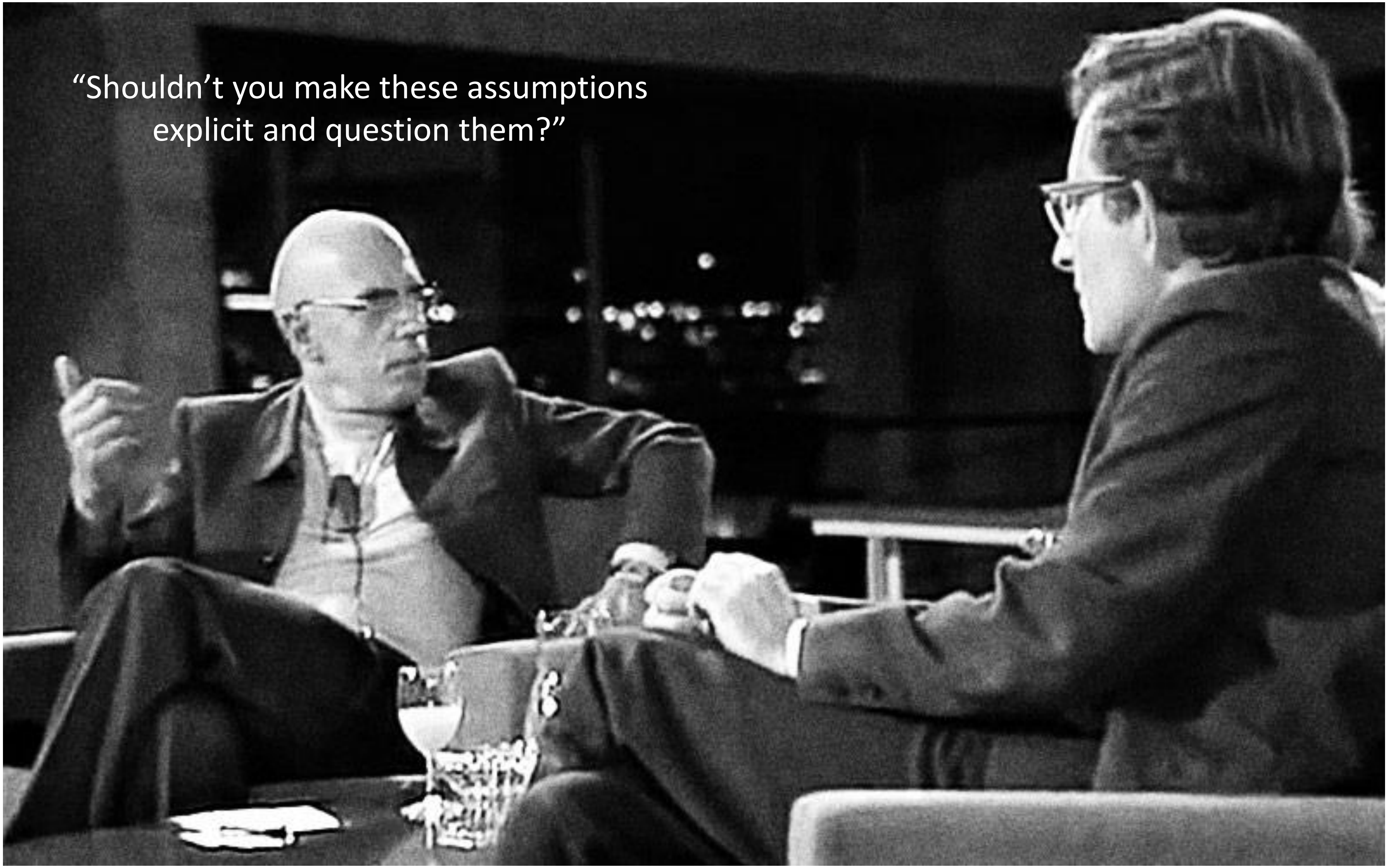
I’ll be honest, I’m not an expert, I found this on the internet somewhere [by Tom Gi, https://www.quora.com/What-did-Michel-Foucault-and-Noam-Chomsky-disagree-about], but it sounds more or less correct to me…

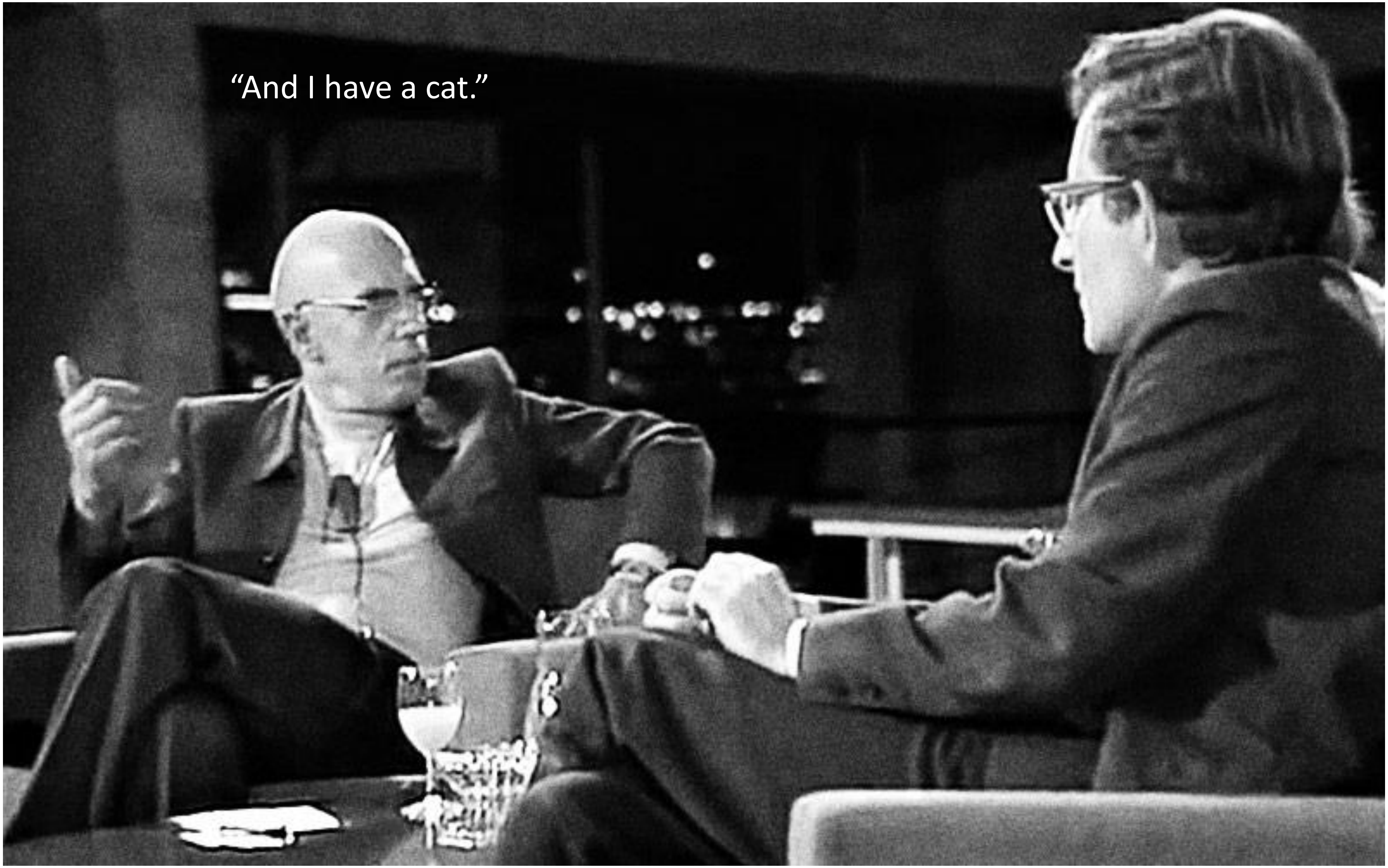

"If we want a
better world we
have to propose
alternatives"


"But surely we have to be precise
about the meanings of terms like
justice and *fairness* – are these not a
construct of an unfair world?"

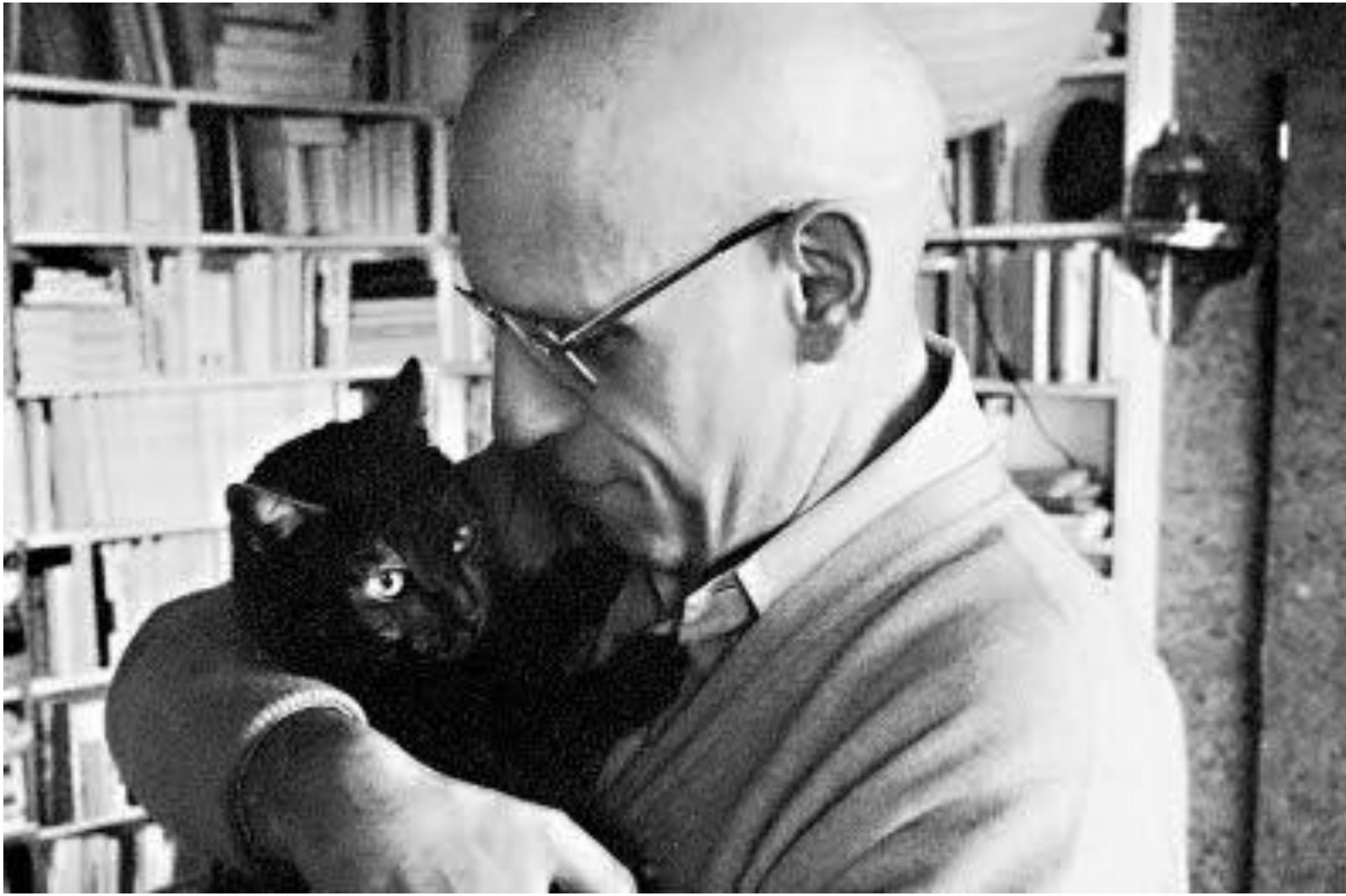

"Don't sit around worrying about NP-hardness. We need cool new heuristics which work well in practice!"


"But how do you define whether one of
your heuristics works well in practice, are
you not unconsciously imposing implicit
assumptions in your model?"


“Shouldn’t you make these assumptions
explicit and question them?”


“And I have a cat.”

Foucault had a cat, he wins



I think both are right

- I’m sure a proper social scientist will tell me this is not possible, but I think both Chomsky and Foucault are right (to some extent)
- It’s important to make (hidden) model assumptions explicit and clear when proposing a better world – but one of the best ways to actually identify these assumptions is to actually try to build it

I think both are right

- In the same way, hardness, lower bounds, rigorous performance guarantees are the "Ying" of heuristics, upper bounds and algorithms that work in practice (the "Yang")
- One without the other has no meaning

I think both are right

- In the same way, hardness, lower bounds, rigorous performance guarantees are the “Ying” of heuristics, upper bounds and algorithms that work in practice (the “Yang”)
- One without the other has no meaning
- The sound of one hand clapping….

I think both are right

- In the same way, hardness, lower bounds, rigorous performance guarantees are the "Ying" of heuristics, upper bounds and algorithms that work in practice (the "Yang")
- One without the other has no meaning
- The sound of one hand clapping….
- Thanks for listening!