A non-data-oriented view of working at Google

Janak Ramakrishnan

Google
http://janak.org/talks/maa-talk.pdf

8 August 2015

Background

- My Ph.D. was in model theory (mathematical logic).
- Specifically, I studied o-minimality, the model theory of "tame" ordered structures.
- Motivating question: why are the real numbers so much easier to understand than the rational numbers? ($\exists y (y^2 = x)$ is a trivial predicate in the reals, less so in the rationals.)
- Questions that I worked on: what do first-order definable linear orders look like in the reals? What about partial orders?
- I did two post-docs, in France and Portugal, before coming to Google.

Background

- My Ph.D. was in model theory (mathematical logic).
- Specifically, I studied o-minimality, the model theory of "tame" ordered structures.
- Motivating question: why are the real numbers so much easier to understand than the rational numbers? ($\exists y(y^2 = x)$ is a trivial predicate in the reals, less so in the rationals.)
- Questions that I worked on: what do first-order definable linear orders look like in the reals? What about partial orders?
- I did two post-docs, in France and Portugal, before coming to Google.

Background

- My Ph.D. was in model theory (mathematical logic).
- Specifically, I studied o-minimality, the model theory of "tame" ordered structures.
- Motivating question: why are the real numbers so much easier to understand than the rational numbers? ($\exists y(y^2 = x)$ is a trivial predicate in the reals, less so in the rationals.)
- Questions that I worked on: what do first-order definable linear orders look like in the reals? What about partial orders?
- I did two post-docs, in France and Portugal, before coming to Google.

Background

- My Ph.D. was in model theory (mathematical logic).
- Specifically, I studied o-minimality, the model theory of "tame" ordered structures.
- Motivating question: why are the real numbers so much easier to understand than the rational numbers? ($\exists y (y^2 = x)$ is a trivial predicate in the reals, less so in the rationals.)
- Questions that I worked on: what do first-order definable linear orders look like in the reals? What about partial orders?
- I did two post-docs, in France and Portugal, before coming to Google.

Background

- My Ph.D. was in model theory (mathematical logic).
- Specifically, I studied o-minimality, the model theory of "tame" ordered structures.
- Motivating question: why are the real numbers so much easier to understand than the rational numbers? ($\exists y(y^2 = x)$ is a trivial predicate in the reals, less so in the rationals.)
- Questions that I worked on: what do first-order definable linear orders look like in the reals? What about partial orders?
- I did two post-docs, in France and Portugal, before coming to Google.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a ... search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a ... search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Google

- Google is a . . . search engine? email service? Video-sharing platform? I'm not sure.
- What's special about Google?
- Google has lots of data.
- What do you do with Really Big Data?
- You can analyze it (machine learning, build models, etc.).
- But to do that, you have to make it accessible.
- Many Google innovations (Bigtable, MapReduce) are about storing/processing data.
- Google scales up its data processing in all realms (logs analysis, for instance).
- That makes Google an interesting place to work, because the bias is always on the side of doing more with data.

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Coding at scale

- Google employs tens of thousands of engineers.
- Many of those engineers spend a lot of time writing code.
- That code all goes into one codebase, where any engineer at Google can see and edit it.
- This is great because it means when an engineer is investigating something, she can dive in and see exactly what the code is, all the way down.
- Imagine being able to drill down and easily see what code is running when an app on your phone crashes, or a webpage fails to load, or a LaTeX file doesn't compile the way you think it should have.
- This is kind of unusual, and it means that Google's tools for dealing with code have to scale better than most tools do.
- The project that I work on is designed to deal with this. It's externally known as Bazel (`http://bazel.io`).

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs

- Bazel's job is to build a binary (an app, a server, both) when you tell it to. This involves some very intricate logic around the gritty details of different computer architectures, language details, etc.
- It also involves a "dependency graph".
- Programs are broken down into units ("libraries"). Each library depends on others.
- These dependency relations give a graph on the libraries.
- Bazel's job at an abstract level is to construct the graph and traverse it.
- And then the hard part of Bazel's job is, after a change, to incrementally modify that graph and rebuild just the parts that need rebuilding.
- This last part is surprisingly hard, so much that many other build tools just give up on it.

Dependency graphs continued

- The dependency graph is a directed acyclic graph (DAG).
- Usually in a dependency graph, if A depends on B, B must be built before A can be built.
- In other words, the graph must be traversed in a manner compatible with the partial order induced by the graph's directed edges.
- Note: this partial order can be extended to a linear order (the *topological order*), but we don't want to execute in topological order (why not?).

Dependency graphs continued

- The dependency graph is a directed acyclic graph (DAG).
- Usually in a dependency graph, if A depends on B, B must be built before A can be built.
- In other words, the graph must be traversed in a manner compatible with the partial order induced by the graph's directed edges.
- Note: this partial order can be extended to a linear order (the *topological order*), but we don't want to execute in topological order (why not?).

Dependency graphs continued

- The dependency graph is a directed acyclic graph (DAG).
- Usually in a dependency graph, if A depends on B, B must be built before A can be built.
- In other words, the graph must be traversed in a manner compatible with the partial order induced by the graph's directed edges.
- Note: this partial order can be extended to a linear order (the *topological order*), but we don't want to execute in topological order (why not?).

Dependency graphs continued

- The dependency graph is a directed acyclic graph (DAG).
- Usually in a dependency graph, if A depends on B, B must be built before A can be built.
- In other words, the graph must be traversed in a manner compatible with the partial order induced by the graph's directed edges.
- Note: this partial order can be extended to a linear order (the *topological order*), but we don't want to execute in topological order (why not?).

Incremental dependency graphs

- An "incremental build" is one in which the dependency graph has already been constructed. You have two nodes, A and B.
- You are interested in the value of A, but you know the value of B has changed. What do you have to do?
- Since A may depend transitively on B, you must recompute some values. Which ones?
- You can recompute the entire downstream transitive closure of B. Then read the value of A.
- You can find the entire upstream transitive closure of A, and then if B is in it, recompute B and propagate those changes.

Incremental dependency graphs

- An "incremental build" is one in which the dependency graph has already been constructed. You have two nodes, A and B.
- You are interested in the value of A, but you know the value of B has changed. What do you have to do?
- Since A may depend transitively on B, you must recompute some values. Which ones?
- You can recompute the entire downstream transitive closure of B. Then read the value of A.
- You can find the entire upstream transitive closure of A, and then if B is in it, recompute B and propagate those changes.

Incremental dependency graphs

- An “incremental build” is one in which the dependency graph has already been constructed. You have two nodes, A and B.
- You are interested in the value of A, but you know the value of B has changed. What do you have to do?
- Since A may depend transitively on B, you must recompute some values. Which ones?
- You can recompute the entire downstream transitive closure of B. Then read the value of A.
- You can find the entire upstream transitive closure of A, and then if B is in it, recompute B and propagate those changes.

Incremental dependency graphs

- An “incremental build” is one in which the dependency graph has already been constructed. You have two nodes, A and B.
- You are interested in the value of A, but you know the value of B has changed. What do you have to do?
- Since A may depend transitively on B, you must recompute some values. Which ones?
- You can recompute the entire downstream transitive closure of B. Then read the value of A.
- You can find the entire upstream transitive closure of A, and then if B is in it, recompute B and propagate those changes.

Incremental dependency graphs

- An "incremental build" is one in which the dependency graph has already been constructed. You have two nodes, A and B.
- You are interested in the value of A, but you know the value of B has changed. What do you have to do?
- Since A may depend transitively on B, you must recompute some values. Which ones?
- You can recompute the entire downstream transitive closure of B. Then read the value of A.
- You can find the entire upstream transitive closure of A, and then if B is in it, recompute B and propagate those changes.

Imperfect incrementality

- Both of these tactics can go horribly wrong.
- What would be ideal? $O(\text{DTC}(B) \cap \text{UTC}(A))$.
- Is that possible?
- In general, the "reachability problem" is solvable in time linear in the size of the graph.
- Similarly, the "transitive closure problem" is solvable in $O(|V||E|)$.
- Too slow!

Imperfect incrementality

- Both of these tactics can go horribly wrong.
- What would be ideal? $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.
- Is that possible?
- In general, the "reachability problem" is solvable in time linear in the size of the graph.
- Similarly, the "transitive closure problem" is solvable in $O(|V||E|)$.
- Too slow!

Imperfect incrementality

- Both of these tactics can go horribly wrong.
- What would be ideal? $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.
- Is that possible?
- In general, the "reachability problem" is solvable in time linear in the size of the graph.
- Similarly, the "transitive closure problem" is solvable in $O(|V||E|)$.
- Too slow!

Imperfect incrementality

- Both of these tactics can go horribly wrong.
- What would be ideal? $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.
- Is that possible?
- In general, the "reachability problem" is solvable in time linear in the size of the graph.
- Similarly, the "transitive closure problem" is solvable in $O(|V||E|)$.
- Too slow!

Imperfect incrementality

- Both of these tactics can go horribly wrong.
- What would be ideal? $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.
- Is that possible?
- In general, the "reachability problem" is solvable in time linear in the size of the graph.
- Similarly, the "transitive closure problem" is solvable in $O(|V||E|)$.
- Too slow!

Imperfect incrementality

- Both of these tactics can go horribly wrong.
- What would be ideal? $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.
- Is that possible?
- In general, the "reachability problem" is solvable in time linear in the size of the graph.
- Similarly, the "transitive closure problem" is solvable in $O(|V||E|)$.
- Too slow!

Imperfect solutions

- Don't actually recompute DTC(B), just "invalidate" it.
- Then traverse UTC(A), but short-circuit any nodes that are not invalidated.
- This still costs $O(\mathrm{DTC}(B))$, but at least computation is just $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.

Imperfect solutions

- Don't actually recompute DTC(B), just "invalidate" it.
- Then traverse UTC(A), but short-circuit any nodes that are not invalidated.
- This still costs $O(\text{DTC}(B))$, but at least computation is just $O(\text{DTC}(B) \cap \text{UTC}(A))$.

Imperfect solutions

- Don't actually recompute DTC(B), just "invalidate" it.
- Then traverse UTC(A), but short-circuit any nodes that are not invalidated.
- This still costs $O(\mathrm{DTC}(B))$, but at least computation is just $O(\mathrm{DTC}(B) \cap \mathrm{UTC}(A))$.

Other imperfect solutions

- Precomputing (caching)?
- Frustrating fact: the graph structure changes frequently.
- Most precomputing isn't effective if you have to do it every time.
- In fact, precomputing ends up being even more expensive.

Other imperfect solutions

- Precomputing (caching)?
- Frustrating fact: the graph structure changes frequently.
- Most precomputing isn't effective if you have to do it every time.
- In fact, precomputing ends up being even more expensive.

Other imperfect solutions

- Precomputing (caching)?
- Frustrating fact: the graph structure changes frequently.
- Most precomputing isn't effective if you have to do it every time.
- In fact, precomputing ends up being even more expensive.

Other imperfect solutions

- Precomputing (caching)?
- Frustrating fact: the graph structure changes frequently.
- Most precomputing isn't effective if you have to do it every time.
- In fact, precomputing ends up being even more expensive.

Google has other interesting problems

- Big data is probably the most important part of Google's mathematically oriented work.
- (Machine learning, linear regressions.)
- A background in statistics+programming is useful, but not essential.
- But, as I've hopefully convinced you, many jobs at Google end up dealing with interesting mathematical/computer science problems.
- Research at Google tends to be very applied – researchers work closely with engineering teams, and engineers also do research.
- Undergraduates with some programming experience are welcome.

Google has other interesting problems

- Big data is probably the most important part of Google's mathematically oriented work.
- (Machine learning, linear regressions.)
- A background in statistics+programming is useful, but not essential.
- But, as I've hopefully convinced you, many jobs at Google end up dealing with interesting mathematical/computer science problems.
- Research at Google tends to be very applied – researchers work closely with engineering teams, and engineers also do research.
- Undergraduates with some programming experience are welcome.

Google has other interesting problems

- Big data is probably the most important part of Google's mathematically oriented work.
- (Machine learning, linear regressions.)
- A background in statistics+programming is useful, but not essential.
- But, as I've hopefully convinced you, many jobs at Google end up dealing with interesting mathematical/computer science problems.
- Research at Google tends to be very applied – researchers work closely with engineering teams, and engineers also do research.
- Undergraduates with some programming experience are welcome.

Google has other interesting problems

- Big data is probably the most important part of Google's mathematically oriented work.
- (Machine learning, linear regressions.)
- A background in statistics+programming is useful, but not essential.
- But, as I've hopefully convinced you, many jobs at Google end up dealing with interesting mathematical/computer science problems.
- Research at Google tends to be very applied – researchers work closely with engineering teams, and engineers also do research.
- Undergraduates with some programming experience are welcome.

Google has other interesting problems

- Big data is probably the most important part of Google's mathematically oriented work.
- (Machine learning, linear regressions.)
- A background in statistics+programming is useful, but not essential.
- But, as I've hopefully convinced you, many jobs at Google end up dealing with interesting mathematical/computer science problems.
- Research at Google tends to be very applied – researchers work closely with engineering teams, and engineers also do research.
- Undergraduates with some programming experience are welcome.

Google has other interesting problems

- Big data is probably the most important part of Google's mathematically oriented work.
- (Machine learning, linear regressions.)
- A background in statistics+programming is useful, but not essential.
- But, as I've hopefully convinced you, many jobs at Google end up dealing with interesting mathematical/computer science problems.
- Research at Google tends to be very applied – researchers work closely with engineering teams, and engineers also do research.
- Undergraduates with some programming experience are welcome.