STATISTICAL PROGRAM ANALYSIS AND SYNTHESIS

Martin Vechev

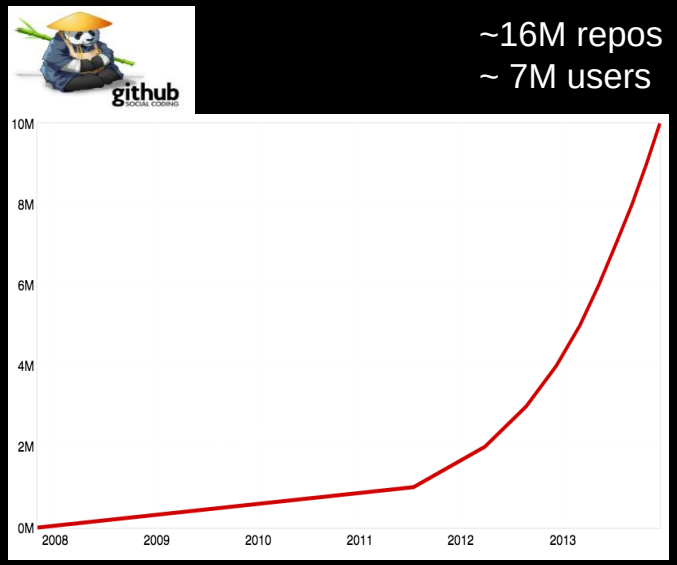
Department of Computer Science
ETH Zurich

Motivation

Unprecedented access to massive codebases
(termed “Big Code” by a recent DARPA initiative)

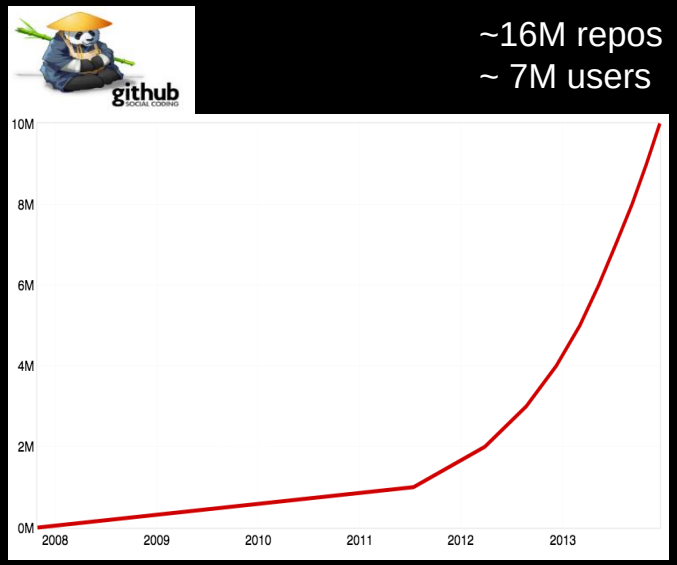
Motivation

~16M repos
~ 7M users



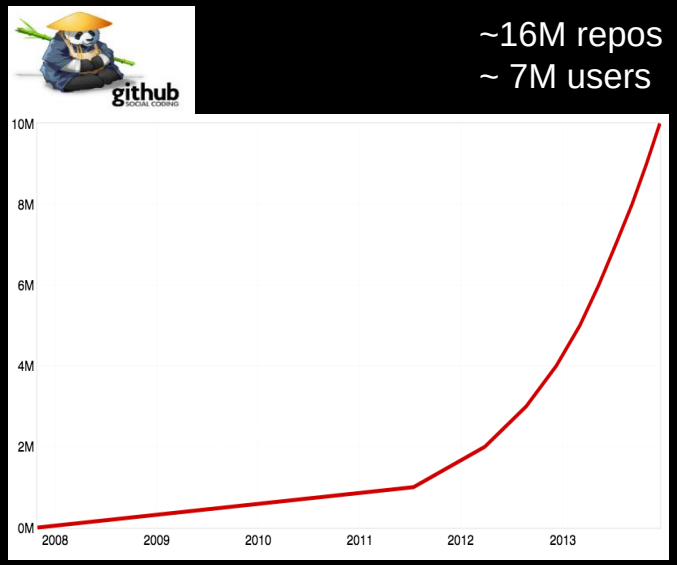
Motivation

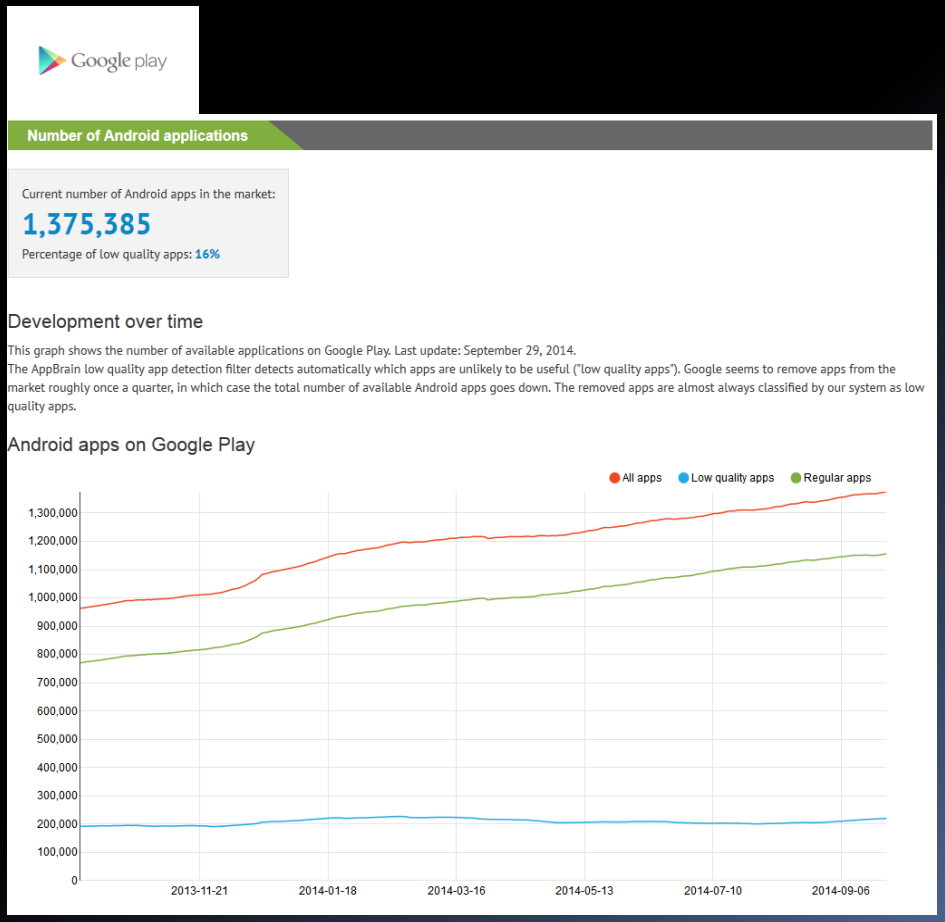
~16M repos
~ 7M users



Motivation

~16M repos
~ 7M users





Vision

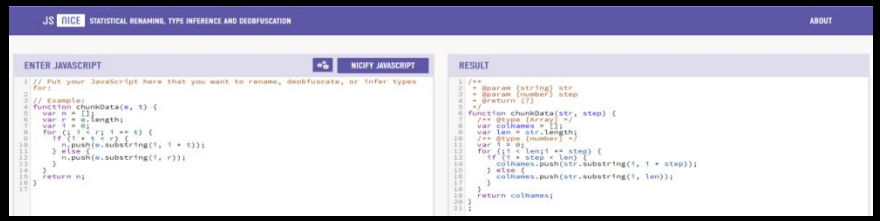
New techniques which solve programming challenges by leveraging availability of massive codebases

General Approach

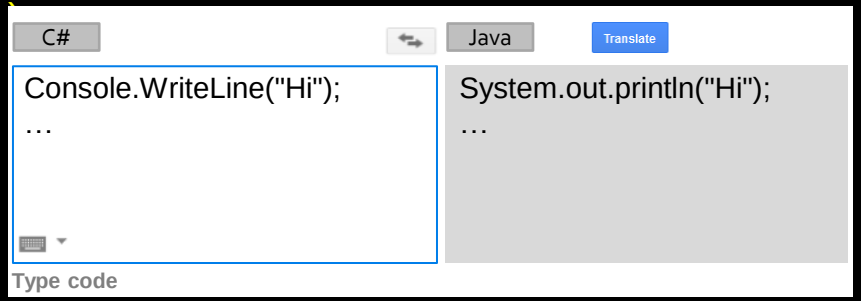
Phrase the problem in a way which allows creative combination of advanced program analysis and powerful machine learning techniques

Statistical Program Analysis and Synthesis

Statistical Prediction of Program Properties, POPL'15
(V. Raychev, M. Vechev, A. Krause)



Statistical PL translation, Onward'14
(S. Karaivanov, V. Raychev, M. Vechev)



Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

Statistical Program Analysis and Synthesis

Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

Slang: Statistical Code Synthesis

```
void exampleMediaRecorder() throws IOException  {
    Camera camera = Camera.open();
    camera.setDisplayOrientation(90);
```

```
    SurfaceHolder holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    MediaRecorder rec = new MediaRecorder();
```

```
    rec.setAudioSource(MediaRecorder.AudioSource.MIC);
    rec.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    rec.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
```

```
}
```

Slang: Statistical Code Synthesis

```
void exampleMediaRecorder() throws IOException {
    Camera camera = Camera.open();
    camera.setDisplayOrientation(90);
    camera.unlock();
    SurfaceHolder holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    MediaRecorder rec = new MediaRecorder();
    rec.setCamera(camera);
    rec.setAudioSource(MediaRecorder.AudioSource.MIC);
    rec.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    rec.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
    rec.setAudioEncoder(1);
    rec.setVideoEncoder(3);
}
```

Slang: Statistical Code Synthesis

```
void exampleMediaRecorder() throws IOException {
    Camera camera = Camera.open();
    camera.setDisplayOrientation(90);
    camera.unlock();
    SurfaceHolder holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    MediaRecorder rec = new MediaRecorder();
    rec.setCamera(camera);
    rec.setAudioSource(MediaRecorder.AudioSource.MIC);
    rec.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    rec.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
    rec.setAudioEncoder(1);
    rec.setVideoEncoder(3);
}
```

sequence not in training data

Slang: Statistical Code Synthesis

```
void exampleMediaRecorder() throws IOException {
    Camera camera = Camera.open();
    camera.setDisplayOrientation(90);
    camera.unlock();
    SurfaceHolder holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    MediaRecorder rec = new MediaRecorder();
    rec.setCamera(camera);
    rec.setAudioSource(MediaRecorder.AudioSource.MIC);
    rec.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    rec.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
    rec.setAudioEncoder(1);
    rec.setVideoEncoder(3);
}
```

sequence not in training data

multiple objects

Slang: Statistical Code Synthesis

```
void exampleMediaRecorder() throws IOException {
    Camera camera = Camera.open();
    camera.setDisplayOrientation(90);
    camera.unlock();
    SurfaceHolder holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    MediaRecorder rec = new MediaRecorder();
    rec.setCamera(camera);
    rec.setAudioSource(MediaRecorder.AudioSource.MIC);
    rec.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    rec.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
    rec.setAudioEncoder(1);
    rec.setVideoEncoder(3);
}
```

sequence not in training data

multiple objects

scalar parameters

Slang: Statistical Code Synthesis

```
void exampleMediaRecorder() throws IOException {
    Camera camera = Camera.open();
    camera.setDisplayOrientation(90);
    camera.unlock();
    SurfaceHolder holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    MediaRecorder rec = new MediaRecorder();
    rec.setCamera(camera);
    rec.setAudioSource(MediaRecorder.AudioSource.MIC);
    rec.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    rec.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
    rec.setAudioEncoder(1);
    rec.setVideoEncoder(3);
}
```

sequence not in training data

prediction is instantaneous

multiple objects

scalar parameters

Statistical Program Analysis and Synthesis

Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

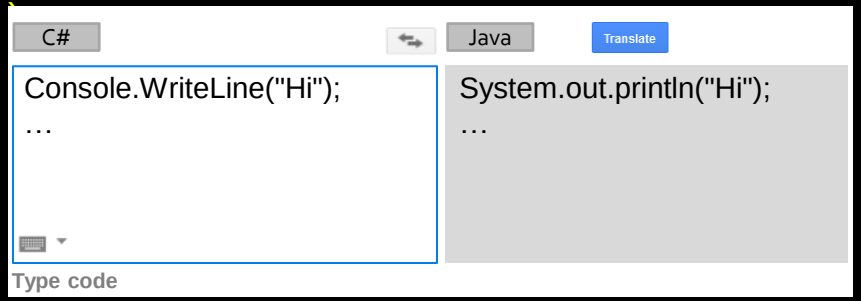
score programs with generative models such as n-gram and recurrent neural networks (RNN)

+

sequence extraction with alias and typestate static analysis

Statistical Program Analysis and Synthesis

Statistical PL translation, Onward'14
(S. Karaivanov, V. Raychev, M. Vechev)



Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

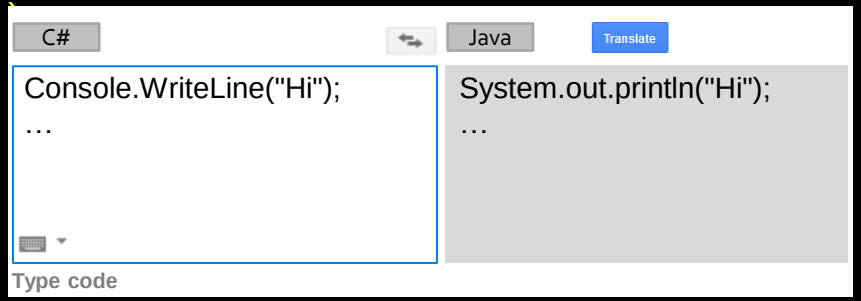
score programs with generative models such as n-gram and recurrent neural networks (RNN)

+

sequence extraction with alias and typestate static analysis

Statistical Program Analysis and Synthesis

Statistical PL translation, Onward'14
(S. Karaivanov, V. Raychev, M. Vechev)



phrase-based statistical machine translation

+

parse prefix grammars during translation

Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

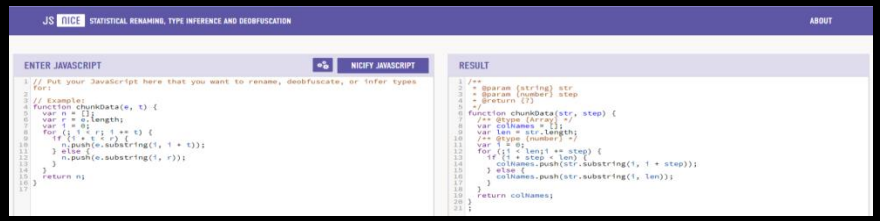
score programs with generative models such as n-gram and recurrent neural networks (RNN)

+

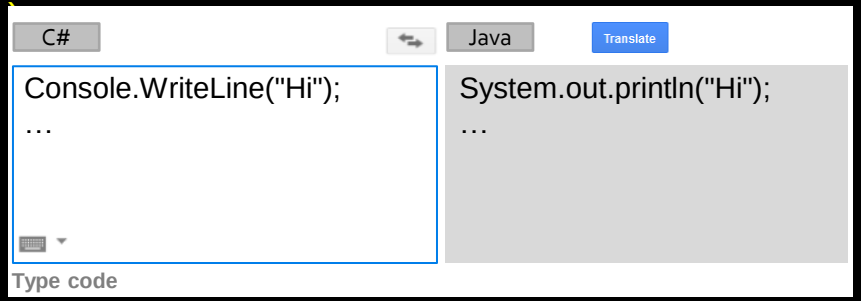
sequence extraction with alias and typestate static analysis

Statistical Program Analysis and Synthesis

Statistical Prediction of Program Properties, POPL'15
(V. Raychev, M. Vechev, A. Krause)



Statistical PL translation, Onward'14
(S. Karaivanov, V. Raychev, M. Vechev)



phrase-based statistical machine translation

+

parse prefix grammars during translation

Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

score programs with generative models such as n-gram and recurrent neural networks (RNN)

+

sequence extraction with alias and typestate static analysis

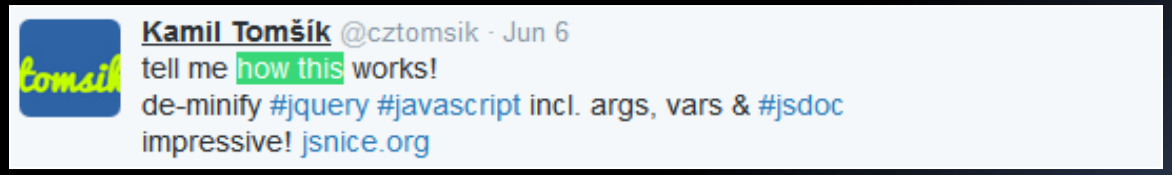
Statistical Program Analysis and Synthesis

Statistical Prediction of Program Properties, POPL'15
(V. Raychev, M. Vechev, A. Krause)

DEMO

JSNice

800+ Tweets (sample below):

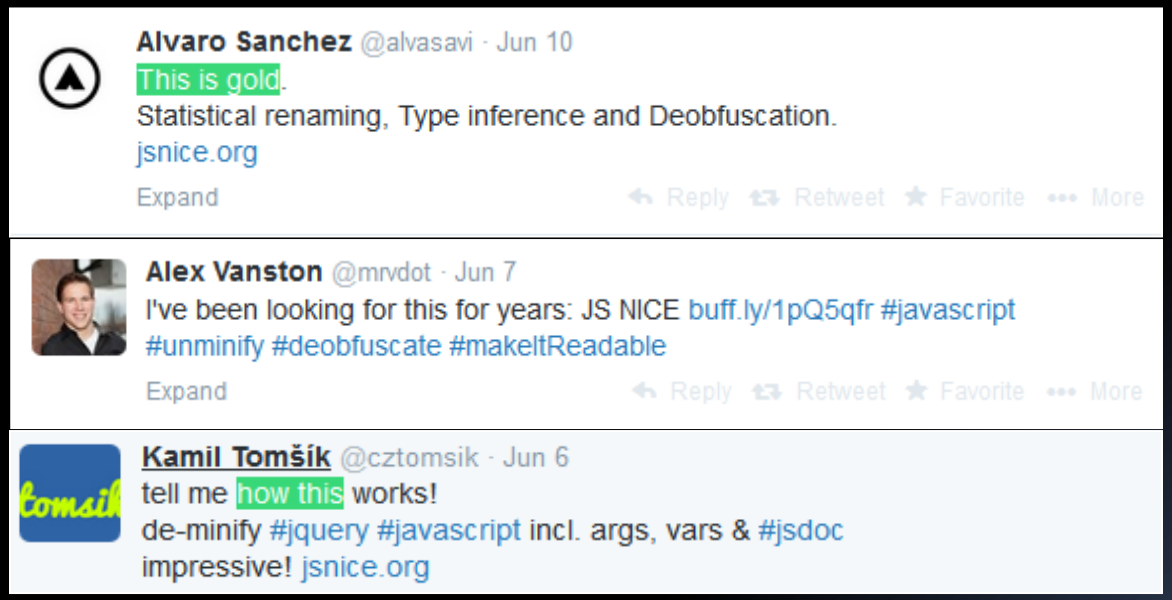


JSNice

800+ Tweets (sample below):

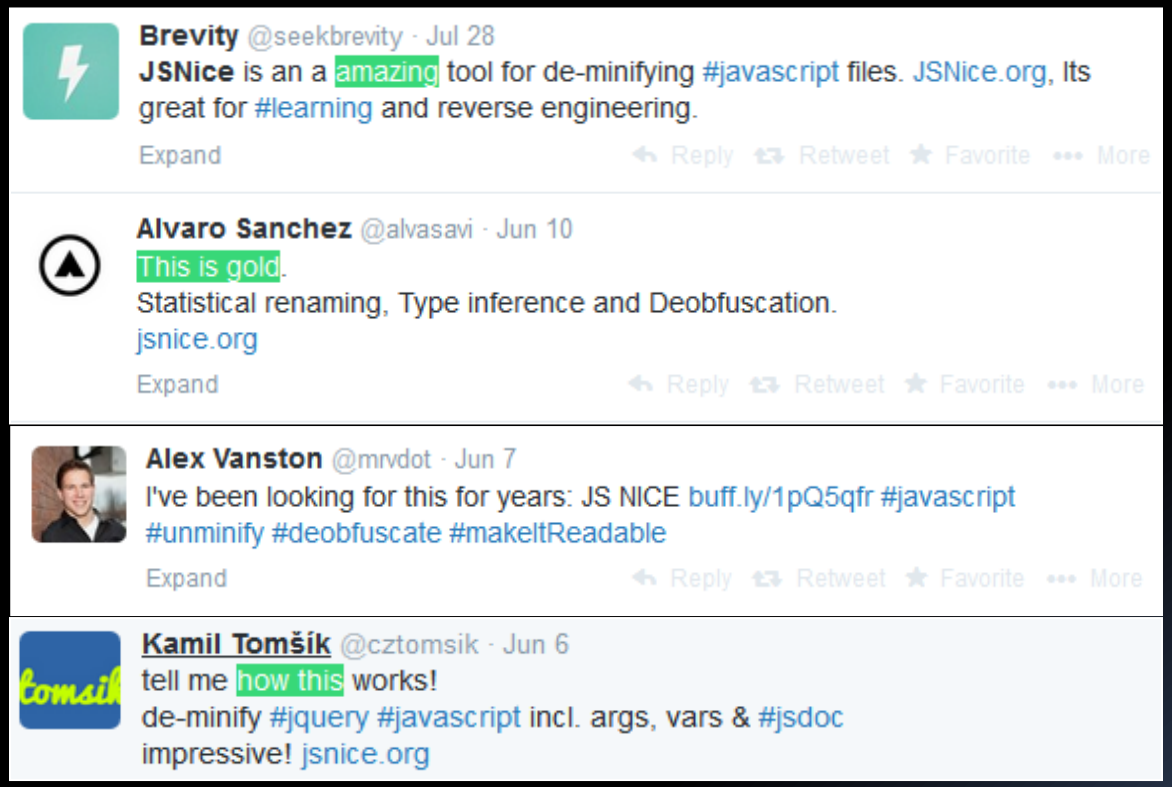
JSNice

800+ Tweets (sample below):



JSNice

800+ Tweets (sample below):



JSNice

800+ Tweets (sample below):

Ingvar Stepanyan @RReverser · Aug 6
JSNice.org became my must-have tool for code deobfuscation.
Expand Reply Retweet Favorite More

Brevity @seekbrevity · Jul 28
JSNice is an a amazing tool for de-minifying #javascript files. JSNice.org, Its great for #learning and reverse engineering.
Expand Reply Retweet Favorite More

Alvaro Sanchez @alvasavi · Jun 10
This is gold.
Statistical renaming, Type inference and Deobfuscation.
jsnice.org
Expand Reply Retweet Favorite More

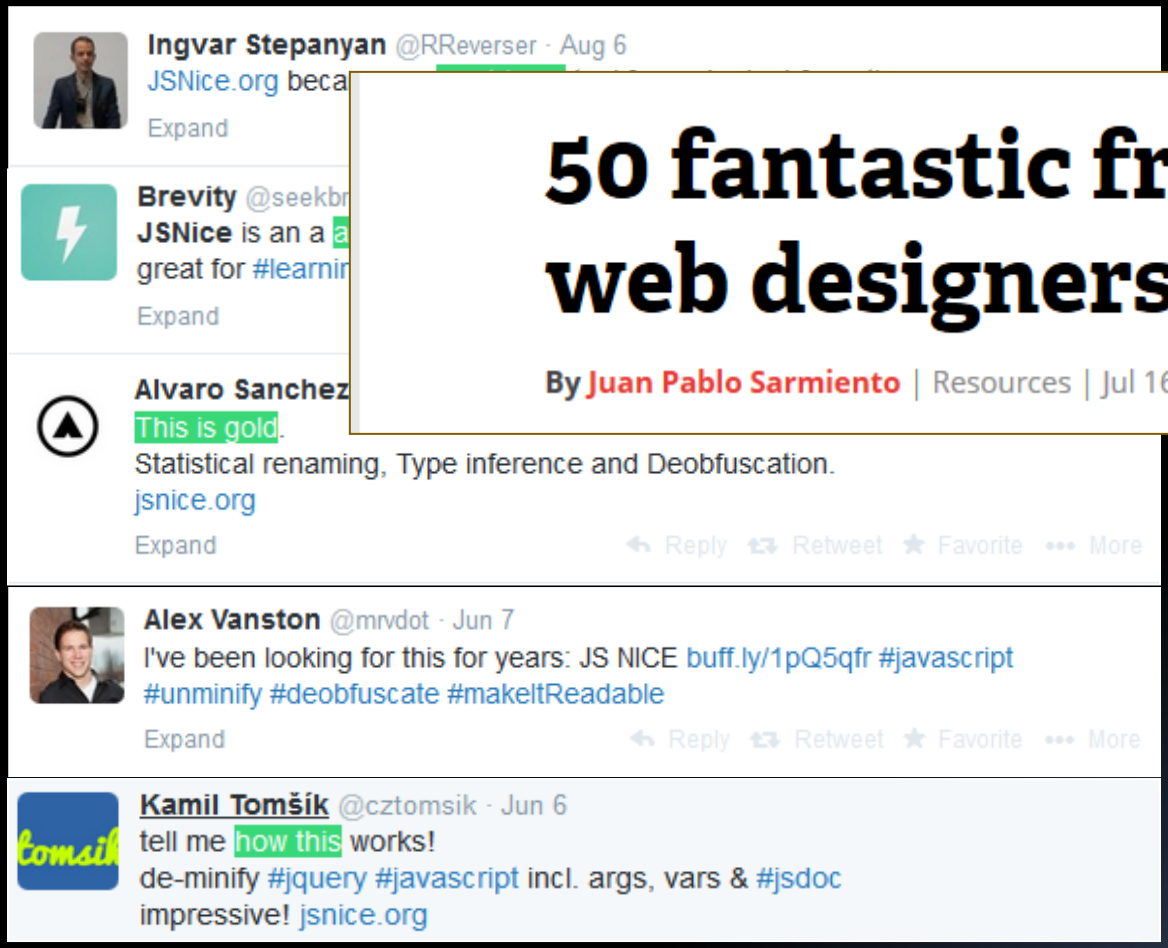
Alex Vanston @mrvdot · Jun 7
I've been looking for this for years: JS NICE buff.ly/1pQ5qfr #javascript #unminify #deobfuscate #makeItReadable
Expand Reply Retweet Favorite More

Kamil Tomšík @cztomsik · Jun 6
tell me how this works!
de-minify #jquery #javascript incl. args, vars & #jsdoc
impressive! jsnice.org

JSNice

- 30,000 users in 1st week
- used in > 170 countries

800+ Tweets (sample below):



Ingvar Stepanyan @RReverser · Aug 6
JSNice.org beca
Expand

Brevity @seekbr
JSNice is an a
great for #learnir
Expand

Alvaro Sanchez
This is gold.
Statistical renaming, Type inference and Deobfuscation.
jsnice.org
Expand Reply Retweet Favorite More

Alex Vanston @mrvdot · Jun 7
I've been looking for this for years: JS NICE buff.ly/1pQ5qfr #javascript #unminify #deobfuscate #makeItReadable
Expand Reply Retweet Favorite More

Kamil Tomšík @cztomsik · Jun 6
tell me how this works!
de-minify #jquery #javascript incl. args, vars & #jsdoc
impressive! jsnice.org

50 fantastic freebies for web designers, July 2014

By Juan Pablo Sarmiento | Resources | Jul 16, 2014

JSNice

- 30,000 users in 1st week
- used in > 170 countries

800+ Tweets (sample below):

Ingvar Stepanyan @RReverser · Aug 6
JSNice.org beca
Expand

Brevity @seekbr
JSNice is an a
great for #learnir
Expand

Alvaro Sanchez
This is gold.
Statistical renaming, Type inference and Deobfuscation.
jsnice.org
Expand Reply Retweet Favorite More

Alex Vanston @mrvdot · Jun 7
I've been looking for this for years: JS NICE buff.ly/1pQ5qfr #javascript #unminify #deobfuscate #makeItReadable
Expand Reply Retweet Favorite More

Kamil Tomšík @cztomsik · Jun 6
tell me how this works!
de-minify #jquery #javascript incl. args, vars & #jsdoc
impressive! jsnice.org

50 fantastic freebies for web designers, July 201

By Juan Pablo Sarmiento | Resources | Jul 16, 20

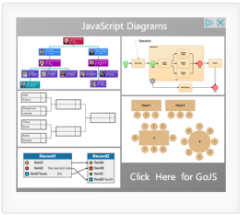
CodeGeekz

COMMENTS

HOME WORDPRESS WEB DESIGN TYPOGRAPHY RESOURCES ARCHIVES CONTACT ADVERTISE

20 Essential Tools for Coders

BY GAVIN IN DEVELOPMENT · RESOURCES — 28.JUL, 2014



Over the last decade, third party tools and free development tools have become an increasingly popular solution for building web applications through a fast and cost-efficient development process. These tools are enriched with various features that turned out to be handy during the development process of an application. Probably this is the reason coders and developers keep looking for free and useful tools which make their work easier and quick.

If you are a coder and looking for some useful development tools then you are at right place, I have gathered **20 Essential Tools for Coders** that would help you out with your development tasks. Following development tools let you write your ideas into code, debug, beautify, store your snippet and help you to simplify your development tasks quickly. We hope you will find the list handy and according to your requirements. Enjoy!

1. JS Nice

JSNice

- 30,000 users in 1st week
- used in > 170 countries

800+ Tweets (sample below):

Ingvar Stepanyan @RReverser · Aug 6
JSNice.org beca
Expand

Brevity @seekbr
JSNice is an a
great for #learnir
Expand

Alvaro Sanchez
This is gold.
Statistical renaming, Type infere
jsnice.org
Expand

Alex Vanston @mrvdot · Jun 7
I've been looking for this for years: JS NICE buff.ly/1pQ5qfr #javascript #unminify #deobfuscate #makeItReadable
Expand Reply Retweet Favorite More

Kamil Tomšík @cztomsik · Jun 6
tell me how this works!
de-minify #jquery #javascript incl. args, vars & #jsdoc
impressive! jsnice.org

50 fantastic freebies for July 2014

AUGUST 28, 2014
JavaScript迷宮救星 — JS nICE

JS NICE

我們網頁前端工程師常常要在網站上用瀏覽器的debug來替網站上的JavaScript檔案除錯，通常那些檔案都已經經過最小化，最佳化，還有變數混淆了，以增進檔案讀取的效率，並防止他人輕易盜用程式碼。不過一旦要除錯，就苦了前端工程師了。

就算現代的瀏覽器如Chrome、Safari與Firefox都支援美化程式碼的功能，讓最小化的程式還原成有排版的情況。但是遇到變數混淆過的程式，面對一堆變數名稱是a、b、c、d的程式，還是會除錯除得很辛苦。

不過網路上有些神奇的工具可以幫助我們不僅是把糊成一團的JS縮排的美美的，甚至它會替我們逆推算已經混淆的變數名稱！

那就是這篇要介紹的JS nICE網站。

這個網站的介面超級乾淨，就像下圖一樣，左邊是複製貼上的JS原始碼，只要按下NICIFY JAVASCRIPT按鈕，經過排版與替換變數的成果就會出現在右邊了！

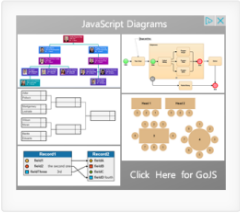
CodeGeekz

COMMENTS
Frici Komesz on: 20 Online Code Editors and Tools for Developers

HOME | WORDPRESS | WEB DESIGN | TYPOGRAPHY | RESOURCES | ARCHIVES | CONTACT | ADVERTISE

20 Essential Tools for Coders

BY GAVIN IN DEVELOPMENT · RESOURCES — 28.JUL, 2014



Over the last decade, third party tools and free development tools have become an increasingly popular solution for building web applications through a fast and cost-efficient development process. These tools are enriched with various features that turned out to be handy during the development process of an application. Probably this is the reason coders and developers keep looking for free and useful tools which make their work easier and quick.

If you are a coder and looking for some useful development tools then you are at right place, I have gathered **20 Essential Tools for Coders** that would help you out with your development tasks. Following development tools let you write your ideas into code, debug, beautify, store your snippet and help you to simplify your development tasks quickly. We hope you will find the list handy and according to your requirements. Enjoy!

1. JS Nice

JSNice

- 30,000 users in 1st week
- used in > 170 countries

800+ Tweets (sample below):

Ingvar Stepanyan @RReverser · Aug 6
JSNice.org beca
Expand

Brevity @seekbr
JSNice is an a
great for #learnir
Expand

Alvaro Sanchez
This is gold.
Statistical renaming, Type infere
jsnice.org
Expand

Alex Vanston @mrvdot · Jun 7
I've been looking for this for years: JS NICE buf
#unminify #deobfuscate #makeItReadable
Expand Reply

Kamil Tomšík @cztomsik · Jun 6
tell me how this works!
de-minify #jquery #javascript incl. args, vars
impressive! jsnice.org

50 fantastic freebies for July 2014

AUGUST 28, 2014
JavaScript迷宮救星 — JS nICE

JS NICE

我們網頁前端工程師常常要在網站上用瀏覽器的debug來替網站上的JavaScript檔案除錯，通常那些檔案都已經經過最小化, 最佳化, 還有變數混淆了，以增進檔案讀取的效率，並防止他人輕易盜用程式碼。不過一旦要除錯，就苦了前端工程師了。

就算現代的瀏覽器如Chrome, Safari
過的程式，面對一堆變數名稱是a, b,
不過網路上有些神奇的工具可以幫助
那就是這篇要介紹的JS nICE網站。
這個網站的介面超級乾淨，就像下圖
變數的成果就會出現在右邊了！

Korben
BLOG LIENS WIKI FOR
LE DÉFI DOMOTIQUE HACKING JEUX VIDÉO RASPBERRY PI ANDROID

1 Jahr gratis Privathaftpflicht-versicherung
Jetzt profitieren

Comment déminifier un code Javascript

CodeGeekz
COMMENTS
HOME WORDPRESS WEB DESIGN TYPOGRAPHY RESOURCES ARCHIVES CONTACT ADVERTISE

20 Essential Tools for Coders

BY GAVIN IN DEVELOPMENT RESOURCES — 28.JUL, 2014

Over the last decade, third party tools and free development tools have become an increasingly popular solution for building web applications through a fast and cost-efficient development process. These tools are enriched with various features that turned out to be handy during the development process of an application. Probably this is the reason coders and developers keep looking for free and useful tools which make their work easier and quick.

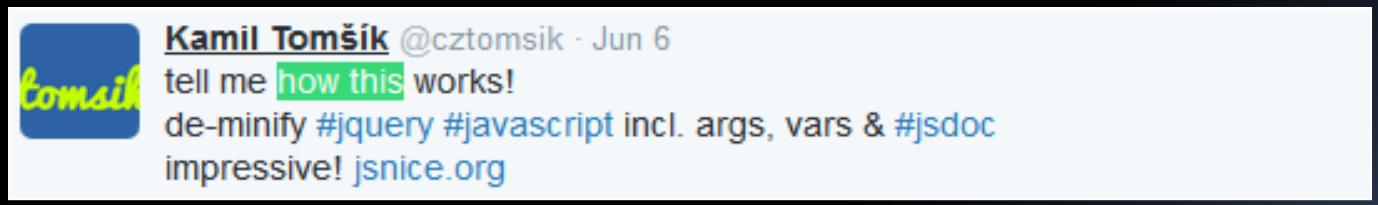
If you are a coder and looking for some useful development tools then you are at right place, I have gathered **20 Essential Tools for Coders** that would help you out with your development tasks. Following development tools let you write your ideas into code, debug, beautify, store your snippet and help you to simplify your development tasks quickly. We hope you will find the list handy and according to your requirements. Enjoy!

1. JS Nice

JSNice

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = 0;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i + step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```



JSNice

$$y = \underset{y \in \Omega}{\operatorname{argmax}} \Pr(y \mid x)$$

JSNice

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = 0;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \operatorname{argmax}_{y \in \Omega} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$$)$$

JSNice

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = 0;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \operatorname{argmax}_{y \in \Omega} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$$)$$

learned from "Big Code"

JSNice

directly captured by a **discriminative** model

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = 0;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \underset{y \in \Omega}{\operatorname{argmax}} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$$)$$

learned from "Big Code"

JSNice

captures **dependence (structure)** between properties

directly captured by a **discriminative** model

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = o;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \underset{y \in \Omega}{\operatorname{argmax}} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = o;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$$)$$

learned from "Big Code"

JSNice

captures **dependence (structure)** between properties

directly captured by a **discriminative** model

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = o;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \underset{y \in \Omega}{\operatorname{argmax}} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = o;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$)$

learned from "Big Code"

constraints on properties

JSNice

captures **dependence (structure)** between properties

directly captured by a **discriminative** model

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = o;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \underset{y \in \Omega}{\operatorname{argmax}} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = o;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$)$

candidate properties

constraints on properties

learned from "Big Code"

JSNice

make a **joint** prediction for most likely properties

captures **dependence (structure)** between properties

directly captured by a **discriminative** model

```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = o;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i+step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

$$= \underset{y \in \Omega}{\operatorname{argmax}} \Pr(y \mid$$

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = o;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

$)$

candidate properties

constraints on properties

learned from "Big Code"

JSNice: Inference

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

JSNice: Inference

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

Unknown properties:

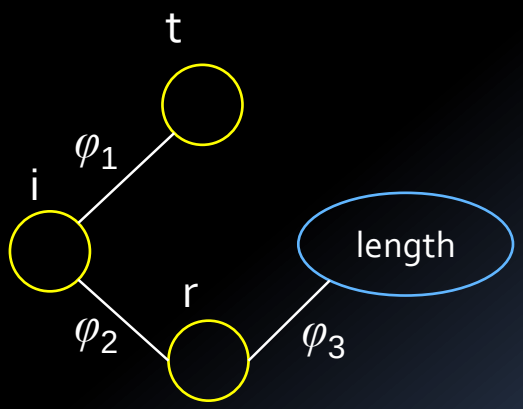
Known properties:

JSNice: Inference

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

Unknown properties:

Known properties:

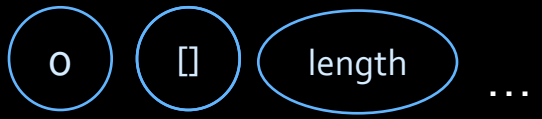


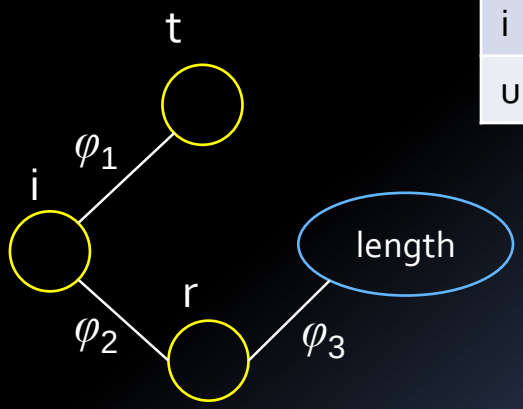
JSNice: Inference

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

Unknown properties:

Known properties:





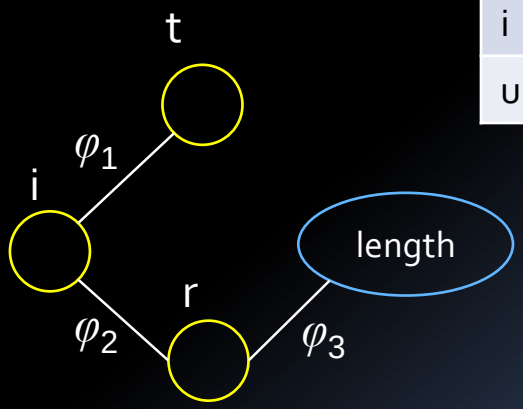
i	t	φ_1
i	step	0.5
j	j	0.4
i	j	0.2
u	q	0.05

JSNice: Inference

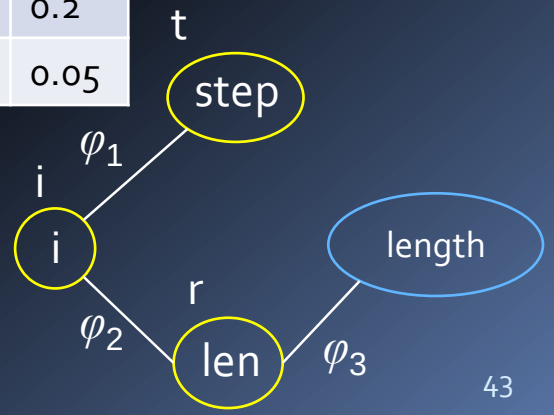
```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

Unknown properties:

Known properties:



i	t	φ_1
i	step	0.5
j	j	0.4
i	j	0.2
u	q	0.05



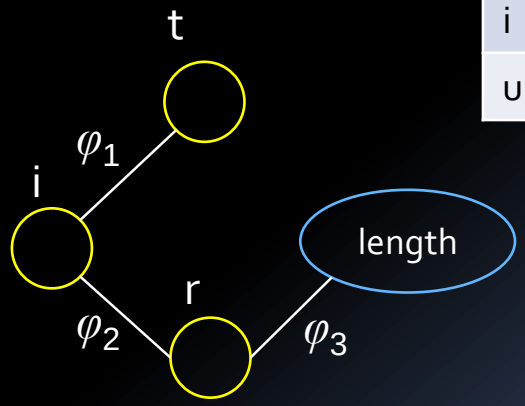
JSNice: Inference

```
function chunkData(e, t)
 var n = [];
 var r = e.length;
 var i = 0;
 for (; i < r; i += t)
   if (i + t < r)
     n.push(e.substring(i, i + t));
  else
     n.push(e.substring(i, r));
 return n;
```

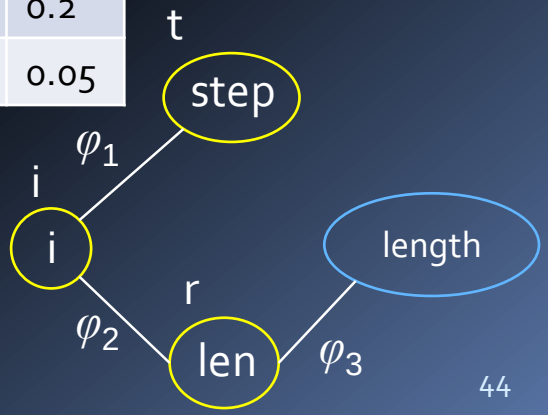
```
function chunkData(str, step)
 var colNames = [];
 var len = str.length;
 var i = 0;
 for (; i < len; i += step)
   if (i + step < len)
     colNames.push(str.substring(i, i + step));
  else
     colNames.push(str.substring(i, len));
 return colNames;
```

Unknown properties:

Known properties:

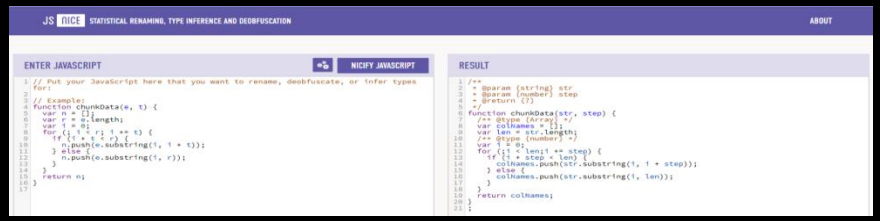


i	t	φ_1
i	step	0.5
j	j	0.4
i	j	0.2
u	q	0.05



Statistical Program Analysis and Synthesis

Statistical Prediction of Program Properties, POPL'15
(V. Raychev, M. Vechev, A. Krause)



structured prediction with discriminative graphical models

+

features based on alias analysis important

Statistical Program Analysis and Synthesis

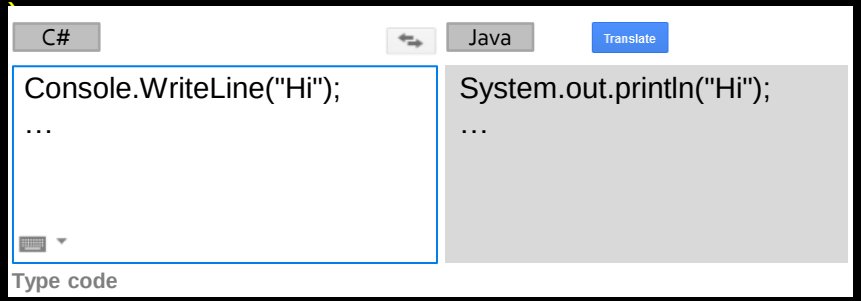
Statistical Prediction of Program Properties, POPL'15
(V. Raychev, M. Vechev, A. Krause)

structured prediction with discriminative graphical models

+

features based on alias analysis important

Statistical PL translation, Onward'14
(S. Karaivanov, V. Raychev, M. Vechev)



phrase-based statistical machine translation

+

parse prefix grammars during translation

Statistical Code Synthesis, PLDI'14
(V. Raychev, M. Vechev, E.Yahav)

```
camera.setDisplayOrientation(90);
camera.unlock();
SurfaceHolder holder = getHolder();
```

score programs with generative models such as n-gram and recurrent neural networks (RNN)

+

sequence extraction with alias and typestate static analysis