A whole new **class** of programming

CS5 overview

CS's building blocks: functions and composition

Loops and iteration

**Designing Data!**

**CS: theory + practice**

whose convenience?
A whole new **class** of programming

**CS5 overview**

**CS's building blocks**: functions and composition

**Loops and iteration**

**Designing Data!**

**CS: theory + practice**
**Wednesday** will be the CS 5 in-class midterm (2\textsuperscript{nd} half)

**Un-warnings:**

concerns? accom.? See me...

five problems, written

*worth 1 hw assignment*

score worries? *Extra* extra-credit in hw 4 +5

**Suggestions:**

go over in-class exercises and hwk problems

create a page of notes, 2-sided is OK

consider small *variations* of the problems – and how they would change the solutions...

*that's our approach*...
Homework 4 preview

- #0: When Algorithms Discriminate...
- #1 ~ lab: Dictionaries, Files, and Benford
- #2: Algorithmic "Authoring"
- #3: vPython (3d graphics)
- (Extra): create-your-own-class...

objects + classes!
Can an Algorithm Hire Better Than a Human?

Hiring and recruiting might seem like some of the least likely jobs to be automated. The whole process seems to need human skills that computers lack, like making conversation and reading social cues.

But people have biases and predilections. They make hiring decisions, often unconsciously, based on similarities that have nothing to do with the job requirements — like whether an applicant has a friend in common, went to the same school or likes the same sports.
When Algorithms Discriminate

Claire Cain Miller @clairecm JULY 9, 2015

The online world is shaped by forces beyond our control, determining the stories we read on Facebook, the people we meet on OkCupid and the search results we see on Google. Big data is used to make decisions about health care, employment, housing, education and policing.

But can computer programs be discriminatory?

Claire Cain Miller @clairecm JUNE 25, 2015

Hiring and recruiting might seem like some of the least likely jobs to be automated. The whole process seems to need human skills that computers lack, like making conversation and reading social cues.

But people have biases and predilections. They make hiring decisions, often unconsciously, based on similarities that have nothing to do with the job requirements — like whether an applicant has a friend in common, went to the same school or likes the same sports.
Homework 4 preview

#0  
*When Algorithms Discriminate...*

#1 ~ lab  
Dictionaries, Files, and *Benford*

#2  
*Algorithmic "Authoring"*

#3  
vPython (3d graphics)

(Extra)  
*create-your-own-class...*
Classes and Objects

An object-oriented programming language allows you to build your own customized types of variables.

(1) A class is a type

(2) An object is one such variable.

There will typically be MANY objects of a single class.
Classes and Objects

(1) A class is a type

(2) An object is one such variable.

(instance)

There will typically be MANY objects of a single class.
Everything in Python is an object!

Its capabilities depend on its class.

functions
"methods"

what's more, you can build your own...
# defining our own Student class

class Student:
    """ a class representing complex student data """

    # the CONSTRUCTOR method (function) [sets initial data]
    def __init__(self, name, yr):
        """ this is the constructor """
        self.name = name
        self.year = yr

    # the "REAPER" method (for printing) [2020 vs '20 ?]
    def __repr__(self):
        """ the not-so-grim reaper? """
        s = self.name + " (" + str(self.year) + ")"
        return s

    # here's a method of our own (not one of Python's)
    def defer(self, numyrs):
        """ defer for a year """
        self.year += numyrs

RS = Student("Robin S.", 2021)  # constructs the object RS
print("RS is", RS)             # __repr__ prints the object
RS.defeer(2)                   # calling a method on/by RS

CA = Student("Cameron A.", 2016)  # constructs another object, CA
CA.defeer(1)                  # CA invokes the defer method
print("CA is", CA)            # prints the object CA
the string class

string object named s

```python
s = str(42)  # This calls the str constructor.
type(s)
<type 'str'>
dir(s)
```

This shows the type of s is str

string methods callable by s

- `__add__`, `__class__`, `__contains__`, `__delattr__`, `__doc__`, `__eq__`, `__format__`, `__ge__`, `__getattribute__`, `__getitem__`, `__getnewargs__`, `__getslice__`, `__gt__`, `__hash__`, `__init__`, `__le__`, `__len__`, `__lt__`, `__mod__`, `__mul__`, `__ne__`, `__new__`, `__reduce__`, `__reduce_ex__`, `__repr__`, `__rmod__`, `__rmul__`, `__setattr__`, `__sizeof__`, `__str__`, `__subclasshook__`, `__formatter_field_name_split`, `__formatter_parser`, `capitalize`, `center`, `count`, `decode`, `encode`, `endswith`, `expandtabs`, `find`, `format`, `index`, `isalnum`, `isalpha`, `isdigit`, `islower`, `isspace`, `istitle`, `isupper`, `join`, `ljust`, `lower`, `lstrip`, `partition`, `replace`, `rfind`, `rindex`, `rjust`, `rpartition`, `rsplit`, `rstrip`, `split`, `splitlines`, `startswith`, `strip`, `swapcase`, `title`, `translate`, `upper`, `zfill`

Let's try some!
import string

This imports the string class

7.1.1. String constants

The constants defined in this module are:

- **string.ascii_letters**
  - The concatenation of the `ascii_lowercase` and `ascii_uppercase` constants described below. This value is not locale-dependent.

- **string.ascii_lowercase**
  - The lowercase letters 'abcdefghijklmnopqrstuvwxyz'. This value is not locale-dependent and will not change.

- **string.ascii_uppercase**
  - The uppercase letters 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'. This value is not locale-dependent and will not change.

- **string.digits**
  - The string '0123456789'.

- **string.hexdigits**
  - The string '0123456789abcdefABCDEF'.

- **string.octdigits**
  - The string '01234567'.

- **string.punctuation**
  - String of ASCII characters which are considered punctuation characters in the C locale.

Lots of constants ("literals") available... Let's check them out...
Let's try some!

\[ L = \text{list}(\ \text{range}(40,43)) \]
are there other classes?

Yes!!!

This is Python's key strength...

Built-in to Python

In external libraries

20 Python libraries you can't live without

1. Requests. The most famous http library written by Kenneth Reitz. It's a must have for every Python developer.

2. Scrapy. If you’re into web scraping then this is a must have library. It’s an open-source library you won’t use any other.

3. Pillow. This is a fork of the Python Imaging Library. It is more user friendly and includes support for more than just images. I have primarily used it in place of PIL (Python Imaging Library).

4. Twisted. The most important tool for any network application developer. It has a very beautiful API and is used by a lot of famous Python projects.

5. SciPy. When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from Ruby to Python.

6. matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyst.

7. Pygame. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.
are there other classes?

Built-in to Python

- 8. Data Types
  - 8.1. datetime — Basic date and time types
  - 8.2. calendar — General calendar-related functions
  - 8.3. collections — Container datatypes
  - 8.4. collections.abc — Abstract Base Classes for Containers
  - 8.5. heapq — Heap queue algorithm
  - 8.6. bisect — Array bisection algorithm
  - 8.7. array — Efficient arrays of numeric values
  - 8.8. weakref — Weak references
  - 8.9. types — Dynamic type creation and names for built-in types
  - 8.10. copy — Shallow and deep copy operations
  - 8.11. pprint — Data printer
  - 8.12. reprlib — Alternate repr() implementation
  - 8.13. enum — Support for enumerations
- 9. Numeric and Mathematical Modules
  - 9.1. numbers — Numeric abstract base classes
  - 9.2. math — Mathematical functions
  - 9.3. cmath — Mathematical functions for complex numbers
  - 9.4. decimal — Decimal fixed point and floating point arithmetic
  - 9.5. fractions — Rational numbers
  - 9.6. random — Generate pseudo-random numbers
  - 9.7. statistics — Mathematical statistics functions
- 10. Functional Programming Modules
  - 10.1. itertools — Functions creating iterators for efficient looping
  - 10.2. functools — Higher-order functions and operations on callables
  - 10.3. operator — Standard operators as functions
- 11. File and Directory Access

In external libraries

20 Python libraries you can’t live without

1. Requests. The most famous http library written by kenneth reitz. It’s a must have for every python developer.

2. Scrapy. If you are involved in web scraping then this is a must have library for you. After using this library you won’t use any other.

3. wxPython. A gui toolkit for python. I have primarily used it in place of tkinter. You will really love it.

4. Pillow. A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.

5. SQLAlchemy. A database library. Many love it and many hate it. The choice is yours.

6. BeautifulSoup. I know it’s slow but this xml and html parsing library is very useful for beginners.

7. Twisted. The most important tool for any network application developer. It has a very beautiful api and is used by a lot of famous python developers.

8. NumPy. How can we learn this very important library? It provides some advance math functionalities to python.

9. SciPy. When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

10. matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyser.

11. Pygame. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2D game development.
are there other classes?

Built-in to Python

We'll introduce two: defaultdict

file

In external libraries

20 Python libraries you can't live without

1. Requests. The most famous http library written by kenneth reitz. It's a must-have for every python developer.

2. Scrappy. If you are involved in webscraping then this is a must-have library for you. After using this library you won't use any other.

3. wxPython. A gui toolkit for python. I have primarily used it in place of tkinter. You will really love it.

4. Pillow. Python Imaging Library. It is more user friendly library for anyone who works with images. Many love it and many hate it. The

5. BeautifulScrap. I know it's slow but this xml and html parsing library is very useful for beginners.

6. Twisted. The most important tool for any network application developer. It has a very beautiful api and is used by a lot of famous python developers.

7. NumPy. How can we leave this very important library? It provides some advance math functionalities to python.

8. SciPy. When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

9. matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyst.

10. Pygame. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.
In Python reading files is no problem...

```
f = open( 'a.txt' )
  # opens the file and calls it f

text = f.read()
  # reads the whole file into the string text

f.close()
  # closes the file (optional)

text
  # 'I like poptarts and 42 and spam.\n
LoW = text.split()
  # text.split() returns a list of each "word"

file class, objects, and methods...
```
CS 5 this week

Using the library

hw4pr1 + hw4pr2

files and the dictionary class

If I had a dictionary, I guess I could look up what it was!

# helper function for problem #1
#

def get_string_from_file( filename ):
    
    """ get_string_from_file should open the file named filename and return all of the text in it as a single string """

    f = open( filename, "r" )  # open filename into the file object f
    text = f.read()            # get all of the text from f as a string
    f.close()                  # close the file f
    return text                # return the string named text
Lists are *sequential* containers:

\[ L = [ 42, 5, 6, 47 ] \]

Elements are looked up by their *location*, or *index*, starting from 0.

Dictionaries are *arbitrary* containers:

\[ d = \{ 3: 47, 1: 5 \} \]

Elements (or *values*) are looked up by a *key* starting anywhere you want! *Keys* don't have to be ints!
Dictionaries are \textit{arbitrary} containers:

\[
d = \{ '\text{rat}' : 1996, '\text{ox}' : 1997 \}\]

elements (or \textit{values}) are looked up by a \textit{key} starting anywhere you want! \textbf{Keys don't have to be ints!}

\textbf{What's up with \texttt{d}'s data here?}

\textbf{Now I see the \texttt{key} to dictionaries' \texttt{value}...}
Dictionaries are *arbitrary* containers:

\[
d = \{ \text{'rat'}: 1996, \text{'ox'}: 1997 \}
\]

elements (or *values*) are looked up by a *key* starting anywhere you want! *Keys* don't have to be ints!

<table>
<thead>
<tr>
<th>Animal</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rat</td>
<td>Feb 19 1996–Feb 06 1997</td>
</tr>
<tr>
<td>Ox</td>
<td>Feb 07 1997–Jan 27 1996</td>
</tr>
<tr>
<td>Tiger</td>
<td>Jan 28 1998–Feb 15 1999</td>
</tr>
<tr>
<td>Rabbit</td>
<td>Feb 16 1999–Feb 04 2000</td>
</tr>
<tr>
<td>Dragon</td>
<td>Feb 05 2000–Jan 23 2001</td>
</tr>
<tr>
<td>Snake</td>
<td>Jan 24 2001–Feb 11 2002</td>
</tr>
<tr>
<td>Horse</td>
<td>Feb 12 2002–Jan 31 2003</td>
</tr>
<tr>
<td>Goat</td>
<td>Feb 01 2003–Jan 21 2004</td>
</tr>
<tr>
<td>Monkey</td>
<td>Jan 22 2004–Feb 08 2005</td>
</tr>
</tbody>
</table>

*Now* I see the *key* to dictionaries' *value*...
Dictionaries are *arbitrary* containers:

\[
z = \{ '\text{rat}' : [1996, 1984, 1972, \ldots] , \\
       '\text{ox}' : [1997, 1985, 1973, \ldots] , \\
       '\text{tiger}' : [1998, \ldots] , \ldots \} 
\]

**What type are the keys?**

**What type are the values?**
Dictionaries are \textcolor{green}{in}!

\[ z = \{ 'rat': [1996, 1984, 1972, \ldots], \]
\[ 'ox': [1997, 1985, 1973, \ldots], \]
\[ 'tiger': [1998, \ldots] \] \]

What if we ask for a key that's not in a dictionary?

\textcolor{red}{Is \ 'dragon' \ key \ in \ z?} \quad \textcolor{orange}{if \ 'dragon' \ in \ z}

\textcolor{red}{Is \ 1969 \ in \ z['dragon']?} \quad \textcolor{orange}{if \ 1969 \ in \ z['dragon']}
Default Dictionaries!!!
In the first line, the `from` keyword is used to import the collections module.

In the second line, the `defaultdict` function is used to create a dictionary `D` with `int` as the default value for any key that is not already present.

In the third line, the `D[42]` operation retrieves the value associated with the key `42` in the dictionary `D`.

How could we change the code so that `D[42]` would be `[]`?
S = "Spam, spam, spam, spam, spam! We <3 spam!"

def word_count(S):
    """word_count returns the number of "words" in S... do you see why "words" is in quotes?! """
    LoW = S.split()
    print("There are", len(LoW), "words")
    return len(LoW)

How many words would this report?
def word_count(S):
    """ word_count returns the number of "words" in S... do you see why "words" is in quotes?! ""
    LoW = S.split()
    print("There are",len(LoW),"words")
    return len(LoW)

S = get_string_from_file('a.txt')

What if we wanted the number of different words in the file?

This would be the author's **vocabulary count**, instead of the total word count.
def vocab_count(S):
    """ dictionaries! """

    DoW = defaultdict(int)

    LoW = S.split()

    for word in LoW:
        DoW[word] += 1

    return DoW

S = "Spam, spam, spam, spam, spam! We <3 spam!"
s = "Spam, spam, spam, spam, spam! We <3 spam!"

low = ['Spam,', 'spam,', 'spam,', 'spam,', 'spam!',
       'spam!', 'We', '<3', 'spam!']

doW = defaultdict( int )

for word in low:
    doW[word] += 1
Vocabulary, anyone?

Shakespeare used **31,534 different words** -- and a grand total of 884,647 words -- counting repetitions (across all of his works) http://www-math.cudenver.edu/~wbriggs/qr/shakespeare.html

---

**Shakespearean coinages**

gust
besmirch
unreal
superscript
watchdog
swagger

**successful**

**affined**

**rooky**

**attasked**

**out-villained**

---

*There's one contemporary author in the Oxford English Dictionary...*

Who?

What word?

---

http://www.pathguy.com/shakeswo.htm
http://www.shakespeare-online.com/biography/wordsinvented.html
Vocabulary, anyone?

Shakespeare used 31,534 different words -- and a grand total of 884,647 words -- counting repetitions (across all of his works)

http://www-math.cudenver.edu/~wbriggs/qr/shakespeare.html

Shakespearean coinages

gust besmirch unreal superscript watchdog swagger

affined rooky attasked out-villained

successful unsuccessful

'Muggle' goes into Oxford English Dictionary

JK Rowling's word for non-wizards - "muggle" -- has made it into the new edition of the Oxford English Dictionary (OED).

The draft definition according to the dictionary’s website says:

- **Muggle**: invented by JK (Joanne Kathleen) Rowling (b. 1965), British author of children's fantasy fiction (see quot. 1997).

In the fiction of JK Rowling: a person who possesses no magical powers. Hence in allusive and extended uses: a person who lacks a particular skill or skills, or who is regarded as inferior in some way.

J. K. Rowling

http://www.pathguy.com/shakeswo.htm
http://www.shakespeare-online.com/biography/wordsinvented.html
Algorithmic authoring?

'Cause somethin' like he left knee and a harp," said he had to the whole school? The shouting and then some strange and Mrs. "Well, I know Hagrid; they spotted handkerchief and get him get rid of course, had a gigantic beet with her," he knew what to all he's

All the sky with the sun in the sun in the church where you're gone Lucy in my eyes. There beneath the girl with an hourglass And then the banker never wears a lot to hold your hand. Can't buy me tight, tight Owww! Love is love I can't hide,

Who is the author? What is the work? What is going on?

This is but ourselves. No, faith, My uncle! O royal bed of confession Of your rue for leave to nature; to this time I should weep for thy life is rotten before he is. have sworn 't. Or my blood. I have closely sent for nine; and unprofitable,

The Senators and the date of a written declaration that Purpose, they shall consist of nine States, shall not, when he shall have such Vacancies. The President pro tempore, in the Desire of a Qualification to the Speaker of the Senate. Article 6. When vacancies by the office upon probable
Markov Models

Techniques for modeling \textit{any} sequence of natural data

\textit{1st-order} Markov Model (defining property)

Each item depends \textit{only} on the one immediately before it.
Our Markov Model

Markov Model

A dictionary!

I like poptarts and 42 and spam. Will I get spam and poptarts for the holidays? I like spam poptarts!

keys  values

'\$': ['I', 'Will', 'I'],
'I': ['like', 'get', 'like'],

'like':

'poptarts': ['and', 'for'],
'and': ['42', 'spam.', 'poptarts'],
'42': ['and'],
'Will': ['I'],

'the':

'spam': ['and', 'poptarts!'],
'get': ['spam'],

'for': ['the']

Why do some words, like holidays, seem missing from the keys?

What type are the keys?
What type are the values?
What are the missing values?
What is the '\$'?
Our Markov Model

Markov Model
A dictionary!

What are the keys?
What are the values?
What are the missing values?
What is the '$'?
Why do some keys seem missing?

{  'predicted': ['I', 'Will', 'I'],
   'I': ['like', 'get', 'like'],
   'like': ['poptarts', 'spam'],
   'poptarts': ['and', 'for'],
   'and': ['42', 'spam', 'poptarts'],
   '42': ['and'],
   'Will': ['I'],
   'the': ['holidays?'],
   'spam': ['and', 'poptarts!'],
   'get': ['spam'],
   'for': ['the']  }

Original file
I like poptarts and 42 and spam. Will I get spam and poptarts for the holidays? I like spam poptarts!
Building d

```
d = defaultdict( list )
pw = '$'

for nw in LoW:
    d[pw] += [nw]

pw = _______
```
Model creation:

1) start with the previous word, \( \text{pw} \) as ' $ '
2) for each next word, \( \text{nw} \), in the list of words, add it in ...
3) then change \( \text{pw} \) to \( \text{nw} \) ...
   
   except if \( \text{nw}[-1] \) was punctuation: then change \( \text{pw} \) to...

Generating text:

1) start with \( \text{pw} \) as the ' $ ' string
2) choose a \( \text{nw} \) that follows \( \text{pw} \), at random.
3) \texttt{print}(nw, \text{end}=' ') \quad \text{(to continue on the same line)}
4) \( \text{pw} \) gets set to either \( \text{nw} \) or ' $ '

what if \( \text{nw}[-1] \) was punctuation?
Markov Models are *generative*!

A key benefit of Markov Models is that they can *generate* feasible data!

**Original file:**

I like poptarts and 42 and spam. Will I get spam and poptarts for the holidays? I like spam poptarts!

**Generated text:**

I get spam poptarts! I like poptarts and 42 and spam. I like spam and 42 and 42 and 42 and spam. **Will I like poptarts and 42 and poptarts and 42 and poptarts and 42 and 42 and poptarts and spam. I get spam and 42 and 42 and...**

gt(d, 200) or gt( cd('hpwhich.txt'), 200 )

Markov Models are *generative*!

A key benefit of Markov Models is that they can *generate* feasible data!

**Original file:**

I like poptarts and 42 and spam. Will I get spam and poptarts for the holidays? I like poptarts and 42 and spam. I like spam and 42 and 42 and spam. Will I like poptarts and 42 and poptarts and 42 and poptarts and 42 and poptarts and spam. I get spam and 42 and 42 and...

**Your task in pr2:** Create a 300-word "CSsay" using a source text of your choice...
WMSCI

Dear Zachary Dodds,

We would like to inform you that we extended to *April 5, 2017* the submission deadline for your potential contribution in the area "Robotics" or any other included in the *21st World Multi-Conference on Systemics, Cybernetics and Informatics: WMSCI 2017* ([http://www.2017iisconf.org/wmsci](http://www.2017iisconf.org/wmsci)), to be held on *July 8 - 11, 2017*, in Orlando, Florida, USA, jointly with:

- The 11th International Multi-Conference on Society, Cybernetics, and Informatics: IMSCI 2017
- The 15th International Conference on Education and Information Systems, Technologies and Applications: EISTA 2017
- The 10th International Multi-Conference on Engineering and Technological Innovation: IMETI 2017

The respective web sites of the above events and the others being jointly organized can be found at the general CFP posted at: [http://www.2017iisconf.org/cfp-summer2017.asp](http://www.2017iisconf.org/cfp-summer2017.asp)

To submit your article, please click the "Authors" tab on the conference website. Submissions for face-to-face and virtual presentations are both accepted.

WMSCI and all its collocated events are being indexed by Elsevier's SCOPUS since 2005. The 2017 proceedings will also be sent to Elsevier's SCOPUS.
<table>
<thead>
<tr>
<th>Title</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMSCI</td>
<td>WMSCI message image! - Here...</td>
</tr>
<tr>
<td>WMSCI 2012 (2)</td>
<td>CFP - Systemics, Cybernetics, and Informatics - Informatics: WMSCI 2012 (<a href="http://www.2012iisconference">http://www.2012iisconference</a> 1/13/12</td>
</tr>
<tr>
<td>WMSCI 2011 (2)</td>
<td>CFP - Deadline Extension - Systemics, Informatics and Cybernetics - Informatics: WMSCI 2011, to be held in Orlando, Florida 2/23/11</td>
</tr>
<tr>
<td>WMSCI 2011 (2)</td>
<td>2nd CFP - Systemics, Informatics and Cybernetics - Informatics: WMSCI 2011, to be held in Orlando, Florida 10/24/10</td>
</tr>
<tr>
<td>WMSCI 2011 (2)</td>
<td>CFP - Systemics, Informatics and Cybernetics - Informatics: WMSCI 2011, to be held in Orlando, Florida 9/9/10</td>
</tr>
<tr>
<td>WM-SCI 2010 (2)</td>
<td>CFP in Artificial Intelligence - Informatics: WMSCI 2010 (<a href="http://www.2010iisconferences.org/wmsci">http://www.2010iisconferences.org/wmsci</a>) 1/17/10</td>
</tr>
<tr>
<td>Sabreen, Zachary (3)</td>
<td>Fwd: Call for Papers and Invited Sessions Proposals - Informatics: WMSCI 2010 &gt; &gt; (<a href="http://www.2010iis">http://www.2010iis</a> 12/13/09</td>
</tr>
<tr>
<td>WMSCI 2010</td>
<td>Call for Papers/Abstracts and Invited Sessions Proposals - Informatics: WMSCI 2010 (June 29th - ... 9/12/09</td>
</tr>
<tr>
<td>AG 2009</td>
<td>Second Call for Papers and Invited Session Proposals - had already presented a quality paper in a presti 1/21/09</td>
</tr>
<tr>
<td>WM-SCI '09 (2)</td>
<td>Call for Papers and Invited Sessions Proposals - org/WMSCI). It will take place in Orlando, Florida, USA 9/4/08</td>
</tr>
<tr>
<td>FIE Conference</td>
<td>Review of paper 1396 - paper at WMSCI 2005, reviewers for conferences need to look out for Markov-g... 4/6/08</td>
</tr>
<tr>
<td>WM-SCI 2008</td>
<td>Last Call for Papers/Abstracts - regards, WMSCI 2008 Secretariat If you wish to be removed from this m 2/29/08</td>
</tr>
<tr>
<td>WM-SCI '08 Announcement</td>
<td>New Call for Papers and Invited Sessions Proposals Deadlines - Proposals for WMSCI 2008: The 12th VI 1/29/08</td>
</tr>
<tr>
<td>Secretariat (2)</td>
<td>Second Call for Papers and Invited Sessions Proposals for WMSCI 2008 - Proposals for WMSCI 2008. 11/4/07</td>
</tr>
<tr>
<td>wmsci2007.sec</td>
<td>Last Call for Papers on Info-CybernEthics 2007 - listed at WMSCI 2007 web page. For submissions c 4/6/07</td>
</tr>
</tbody>
</table>
WMSCI 2005

Rooter: A Methodology for the Typical Unification of Access Points and Redundancy

Jeremy Stribling, Daniel Aguayo and Maxwell Krohn

http://pdos.csail.mit.edu/scigen/

Markov-generated submission accepted to WMSCI 2005
this was *more* than a first-order model...
and, scene...!
in costume