ExCAPE
Expeditions in Computer Augmented Program Engineering

Rajeev Alur, Ras Bodik, Jeff Foster, Bjorn Hartmann, Lydia Kavraki, Hadas Kress-Gazit, Stephane LaFortune, Boon Loo, P. Madhusudan, Milo Martin, George Pappas, Alberto Sangiovanni-Vincentelli, Sanjit Seshia, Armando Solar-Lezama, Paulo Tabuada, Stavros Tripakis, Moshe Vardi, Steve Zdancewic

Cornell, Maryland, Michigan, MIT, Penn, Rice, UC Berkeley, UCLA, UIUC

Reverse Site Visit, National Science Foundation, December 2011
Software: Enabling Technology with a Caveat

Software Inside!

Software ➔ New features, Automation, Customization, Flexibility

Software ➔ Bugs, Cost overruns, Cancelled projects
Software: Enabling Technology with a Caveat

Software → New features, Automation, Customization, Flexibility

Software → Bugs, Cost overruns, Cancelled projects

Grand challenge: Transform technology for software development
Software Design Methodology

- What has changed:
  - Programming languages
  - Libraries
  - Verification technology

- What has not changed:
  - Programming is done by experts
  - Fully specified by conventional programming
  - Verification phase is distinct from design
Software Design Methodology

- **What has changed:**
  - Programming languages
  - Libraries
  - Verification technology

- **What has not changed:**
  - Programming is done by experts
  - Fully specified by conventional programming
  - Verification phase is distinct from design

Can we leverage modern analysis tools and increased computing power to revolutionize the task of programming?

Inspiration: Recent innovations in synthesis illustrated by 3 projects
Err = 0.0;
for(t = 0; t<T; t+=dT){
    if(stage==STRAIGHT){
        if(t > ??) stage= INTURN;
    }
    if(stage==INTURN){
        car.ang = car.ang - ??;
        if(t > ??) stage= OUTTURN;
    }
    if(stage==OUTTURN){
        car.ang = car.ang + ??;
        if(t > ??) break;
    }
    simulate_car(car);
    Err += check_collision(car);
}
Err += check_destination(car);
Err = 0.0;
for(t = 0; t<T; t+=dT){
    if(stage==STRAIGHT){
        if(t > ??) stage= INTURN;
    }
    if(stage==INTURN){
        car.ang = car.ang - ??;
        if(t > ??) stage= OUTTURN;
    }
    if(stage==OUTTURN){
        car.ang = car.ang + ??;
        if(t > ??) break;
    }
    simulate_car(car);
    Err += check_collision(car);
}
Err += check_destination(car);
Err = 0.0;
for(t = 0; t<T; t+=dT){
    if(stage==STRAIGHT){
        if(t > ??) stage= INTURN;
    }
    if(stage==INTURN){
        car.ang = car.ang - ??;
        if(t > ??) stage= OUTTURN;
    }
    if(stage==OUTTURN){
        car.ang = car.ang + ??;
        if(t > ??) break;
    }
    simulate_car(car);
    Err += check_collision(car);
}
Err += check_destination(car);
Sketch: Program completion

Ref: Solar-Lezama et al (PLDI 2010)

Err = 0.0;
for(t = 0; t<T; t+=dT){
  if(stage==STRAIGHT){
    if(t > ??) stage= INTURN;
  }
  if(stage==INTURN){
    car.ang = car.ang - ??;
    if(t > ??) stage= OUTTURN;
  }
  if(stage==OUTTURN){
    car.ang = car.ang + ??;
    if(t > ??) break;
  }
  simulate_car(car);
  Err += check_collision(car);
}
Err += check_destination(car);

Enables programmers to focus on high-level solution strategy
## QuickCode: Programming by Examples

Ref: Gulwani (POPL 2011)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>(425)-706-7709</td>
<td>425-706-7709</td>
</tr>
<tr>
<td>510.220.5586</td>
<td>510-220-5586</td>
</tr>
<tr>
<td>1 425 235 7654</td>
<td>425-235-7654</td>
</tr>
<tr>
<td>425 745-8139</td>
<td>425-745-8139</td>
</tr>
</tbody>
</table>

- Infers desired Excel macro program
- Iterative: user gives examples and corrections
- Being incorporated in next version of Microsoft Excel
QuickCode: Programming by Examples

Ref: Gulwani (POPL 2011)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>(425)-706-7709</td>
<td>425-706-7709</td>
</tr>
<tr>
<td>510.220.5586</td>
<td>510-220-5586</td>
</tr>
<tr>
<td>1425 235 7654</td>
<td>425-235-7654</td>
</tr>
<tr>
<td>425 745-8139</td>
<td>425-745-8139</td>
</tr>
</tbody>
</table>

Infers desired Excel macro program
Iterative: user gives examples and corrections
Being incorporated in next version of Microsoft Excel

Enables non-programmers to program interactively
Paraglide: From Sequential to Parallel Code
Ref: Vechev et al (POPL 2010)

Sequential Program

```c
bool add(int key){
    atomic
    Entry *pred,*curr,*entry
    locate(pred,curr,key);
    k = (curr->key == key)
    if (k) return false
    entry = new Entry()
    entry->next = curr
    pred->next = entry
    return true
}
```

Minimal Synchronization

```c
bool add(int key) {
    Entry *pred,*curr,*entry
    restart:
    locate(pred,curr,key)
    k = (curr->key == key)
    if (k) return false
    entry = new Entry()
    entry->next = curr
    val = CAS(&pred->next,<curr,0>,<entry,0>)
    if (!val) goto restart
    return true
}
```

Architecture Description

- **Target:** Highly concurrent work queue in C/C++
- **Infers** minimal number of fences needed for synchronization
- **Unexpected, correct, minimal solutions** now deployed in IBM
Paraglide: From Sequential to Parallel Code

Ref: Vechev et al (POPL 2010)

**Sequential Program**

```c
bool add(int key) {
    Entry *pred,*curr,*entry
    locate(pred,curr,key);
    k = (curr->key == key)
    if (k) return false
    entry = new Entry()
    entry->next = curr
    pred->next = entry
    return true
}
```

**Minimal Synchronization**

```c
bool add(int key) {
    Entry *pred,*curr,*entry
    restart:
    locate(pred,curr,key)
    k = (curr->key == key)
    if (k) return false
    entry = new Entry()
    entry->next = curr
    val= CAS(&pred->next,<curr,0>,<entry,0>)
    if (!val) goto restart
    return true
}
```

**Architecture Description**

- Target: Highly concurrent work queue in C/C++
- Infers minimal number of fences needed for synchronization
- Unexpected, correct, minimal solutions now deployed in IBM

Enables programmers to meet new programming challenges
ExCAPE Vision

Harnessing computation to transform programming:
Programming made easier, faster, cheaper
Key enabler for next-generation software applications
ExCAPE Design Solution

- Designer expresses “what”, possibly using multiple input formats
- Synthesizer discovers new artifacts via integration and completion
- Synthesizer solves computationally demanding problems using advanced analysis tools
- Interactive iterative design
- Integrated formal verification
Challenge Problems

- Representative of complexity: cyber-physical systems on networked, multi-core platforms
- Concrete design problems to guide tools and methodology
- Multiple challenge problems to avoid domain-specific solutions
Proposed Research

In each challenge area,

- Identify a concrete design problem for which new solutions can enable new applications
- Identify most promising synthesis-based solution strategies
- Develop theoretical foundations and algorithmic advances
- Build tools and prototypes
- Evaluate tools for scalability, user interaction, and programmer productivity
- Refine and advance computational/methodological solutions and tools
Multicore Protocols: ExCAPE Design Solution
Multicore Protocols: Research Questions

- How to consistently integrate (partial) state machines, example scenarios, and temporal-logic requirements?
- How to suggest potential fixes?
- What’s a good programming notation for multi-modal specifications?
- How to program synthesis engine with completion strategies specific to a problem domain (e.g. cache coherence)?
- How to address scalability?
- How to evaluate and measure impact on programmer productivity?
Impacting Industrial Practice

- **Keys to transitioning academic research to industrial practice**
  1. Market pull and industrial interest
  2. Algorithmic advances and computational tools
  3. Methodology for integration in design cycle

- **Our plan:** Advance computational tools and methodology, and demonstrate benefits on meaningful case studies

- **Collaborators:**
  - Chitta (Willow Garage), Gulwani (Microsoft), Vechev (IBM)

- **Advisory Board:**
  - Fix (Intel), Godbole (Honeywell), Kuehlmann (Coverity), Lee (Microsoft), Wegman (IBM), Zave (AT&T)
Education and Outreach

- Annual workshop
  - Academic and industrial participants
- Summer school
  - Integrative and multi-disciplinary training
- Synthesis competition
  - Benchmarks and tool evaluation
- Undergraduate education
  - Course modules for CS and CE courses
- Attracting high-school students to CS and Engineering
  - Programming is not equal to coding
  - Projects in robotics
  - Collaboration with existing high-school programs at PI institutions
ExCAPE Vision

Harnessing computation to transform programming:
Programming made easier, faster, cheaper
Key enabler for next-generation software applications