Correctness Enhancement as a Pervasive SE Paradigm

Ali Mili, NJIT
Third Spring Festival Workshop
Karuizawa, March 2017
Acknowledgements

• In collaboration with
  – M. Frias (Argentina),
  – J. Desharnais (CDN),
  – W. Ghardallou (TN),
  – N. Diallo (US)

• Thanks to Professor Zhenjiang Hu
  – Invitation to the Spring Festival Workshop

• Work partially supported by
  – NSF (Grant DGE 1565478)
  – Qatar Foundation (NPRP04-1109-1-174)
Overview

- Premise/ Agenda
- Relational Mathematics for Absolute Correctness
- Relational Mathematics for Relative Correctness
- Correctness Enhancement: A Pervasive Paradigm
- Implications
- Conclusion
Premise/ Agenda

Relative correctness:

• The property of a program $P'$ to be more-correct than a program $P$ with respect to a specification $R$.

• Whereas absolute correctness divides candidate programs into two classes
  – Correct, incorrect,

relative correctness ranks candidate programs on a partial ordering.
  – Maximal elements: the correct programs.
Premise/ Agenda

Correctness Enhancement:

• The process of transforming a program $P$ to make it more-correct than it is with respect to a specification $R$.
  – The new version may still be incorrect.
Premise/ Agenda

Correctness Enhancement Pervades SE
- Program Construction
- Corrective Maintenance
- Software Repair
- Adaptive maintenance
- Whitebox Reuse
  - Search
  - Adaptation
- Programming for Reliability.
- Program Merger.
- Program Upgrade.
- Test Driven Programming/ Extreme Programming
Premise/ Agenda

Agenda:
• Explore the mathematics of relative correctness/correctness enhancement.
• Discuss how and to what extent relative correctness pervades software engineering.
• Contemplate implications.
Overview

- Premise/ Agenda
- **Relational Mathematics for Absolute Correctness**
- Relational Mathematics for Relative Correctness
- Correctness Enhancement: A Pervasive Paradigm
- Implications
- Conclusion
Relational Mathematics of Absolute Correctness

- **Space**: Set $S$,
- **Specification**: Relation $R$ on $S$.
- **Program**: Function $P$ on $S$, mapping initial states to final states.
- **Refinement**: $(R' \sqsupseteq R) \iff RL \cap R' \cap (R \cup R') = R'$
- Program $P$ is (absolutely) correct with respect to specification $R$ if and only if: $P$ refines $R$. 
Relational Mathematics of Absolute Correctness

Figure 1: $R' \sqsubseteq R$
Relational Mathematics of Absolute Correctness

• Refinement: Partial Ordering.

• Lattice Properties:
  – Any two specifications have a meet (greatest lower bound):
    \[(R \cap R') = RL \cap R' \cap (R \cup R')\]

  – Two specifications R and R’ have an upper bound only if they are compatible:
    \[RL \cap R'L = (R \cap R')L\]

  – The join (least upper bound) of two compatible specifications is given by the formula:
    \[(R \cup R') = \overline{RL} \cap R \cup \overline{RL} \cap R' \cup (R \cap R')\]
Relational Mathematics of Absolute Correctness

Compatibility Condition:

• R: \( x + 2 \leq x' \leq x + 6 \)
• R': \( x + 4 \leq x' \leq x + 8 \)
  - Join: \( x + 4 \leq x' \leq x + 6 \)

• R: \( x + 2 \leq x' \leq x + 6 \)
• R': \( x + 8 \leq x' \leq x + 12 \)
  - No join.

Figure 1: Lattice Operators
Relational Mathematics of Absolute Correctness

Join of two specifications $R$ and $R'$:
- Sum of their requirements.
- Defined only if they are compatible

Meet of two specifications $R$ and $R'$:
- Common requirements.

Figure 1: Lattice Operators
Relational Mathematics of Absolute Correctness

Subtracting Specifications:
- $7 - 4: \ x, \text{ s.t. } x + 4 = 7$.
- Also: $\min_x: \ x + 4 \geq 7$.

Likewise:
- If $R'$ refines $R$, we define:

\[
R' \ominus R = \min_x : R \cup X \supseteq R'.
\]

\[
R' \ominus R = (R' \cap \overline{RL}) \cup (R \cap \overline{R'}) \cap \overline{R}.
\]
Relational Mathematics of Absolute Correctness

Since we have subtractions, why not distances:

- Real numbers: \( d(x, y) = \max(x, y) - \min(x, y) \).

Specifications:

\[
\delta(R, R') = (R \sqcup R') \ominus (R \cap R').
\]

- The distance is a specification, not a number.
- Satisfies All the axioms of distance.
- Enables us to compare proximity.
Relational Mathematics of Absolute Correctness

\[ \delta(R, Q') \subseteq \delta(R, Q) \]
Overview

- Premise/ Agenda
- Relational Mathematics for Absolute Correctness
- Relational Mathematics for Relative Correctness
- Correctness Enhancement: A Pervasive Paradigm
- Implications
- Conclusion
Relational Mathematics for Relative Correctness
We are given:

- Specification $R$ on space $S$.
- Two deterministic programs $P$ and $P'$.
- Competence domain of $P$ with respect to $R$: set of initial states on which $P$ satisfies $R$.
  - $\text{dom}(R \cap P)$.
  - $(R \cap P)L$.
- $P'$ (strictly) more-correct than $P$ with respect to $R$:
  - Competence domain of $P'$ (proper) superset of that of $P$ (wrt $R$).
Relational Mathematics for Relative Correctness

Figure 4: \( P' \sqsupseteq_R P \), Deterministic Specifications
Relational Mathematics for Relative Correctness

Is this definition any good? How do we know?

- Relative Correctness: Reflexive, transitive, not antisymmetric.
  - Does not necessarily duplicate correct behavior.

- Culminates in Absolute Correctness.
  
- Logically Implies Reliability
  
- Pointwise Refinement

\[ P' \sqsupseteq R \Rightarrow (\forall P, P' \sqsupseteq_R P). \]

\[ P' \sqsupseteq_R P \Rightarrow (\forall \theta : \rho^\theta_R(P') \geq \rho^\theta_R(P)). \]

\[ P' \sqsupseteq P \Leftrightarrow (\forall R : P' \sqsupseteq_R P). \]
Relational Mathematics for Relative Correctness

\[ P' \supseteq P \]

\[ \forall \theta \]

\[ \forall R \]

\[ \int_{\text{dom}(P' \cap P')} \theta(s) \, ds \geq \int_{\text{dom}(P)} \theta(s) \, ds \]

\[ \rho_h^\theta(P') \geq \rho_h^\theta(P) \]

\[ P' \supseteq_r P \]

\[ \forall R \]

\[ \forall \theta \]
Relational Mathematics for Relative Correctness

\[ R = \{(s, s') | s^2 \leq s' \leq s^3\} \]

p0: {abort}. \( P_0 = \phi. \) \( CD_0 = \emptyset. \)

p1: {s=0;}. \( P_1 = \{(s, s') | s' = 0\}. \) \( CD_1 = \{0\}. \)

p2: {s=1;}. \( P_2 = \{(s, s') | s' = 1\}. \) \( CD_2 = \{1\}. \)

p3: {s=2*s**3-8;}. \( P_3 = \{(s, s') | s' = 2s^3 - 8\}. \) \( CD_3 = \{2\}. \)

p4: {skip;}. \( P_4 = I. \) \( CD_4 = \{0, 1\}. \)

p5: {s=2*s**3-3*s**2+2;}. \( P_5 = \{(s, s') | s' = 2s^3 - 3s^2 + 2\}. \) \( CD_5 = \{1, 2\}. \)

p6: {s=s**4-5*s;}. \( P_6 = \{(s, s') | s' = s^4 - 5s\}. \) \( CD_6 = \{0, 2\}. \)

p7: {s=s**2;}. \( P_7 = \{(s, s') | s' = s^2\}. \) \( CD_7 = S. \)

p8: {s=s**3;}. \( P_8 = \{(s, s') | s' = s^3\}. \) \( CD_8 = S. \)

p9: {s=(s**2+s**3)/2;}. \( P_9 = \{(s, s') | s' = \frac{s^2 + s^3}{2}\}. \) \( CD_9 = S. \)
Relational Mathematics for Relative Correctness

Whereas absolute correctness divides candidate programs into two categories, relative correctness defines a richer partial ordering whose maximal elements are absolutely correct.
Relational Mathematics for Relative Correctness

Relative Correctness of Non Deterministic Programs

- What is a Non Deterministic Program?
- Why do we need to model such programs?

\[ R = \{(s, s') | a[f'] = x \land 1 \leq f' \leq N \land (\forall h : f' < h \leq N : a[h] \neq x)\}. \]

- Place in \( f \) the largest index where \( x \) is located in \( a \) (\( x \) is known to be in \( a \)).
Relational Mathematics for Relative Correctness

- **Program** $P$:

  ```
  int k; int z; z=1; k=1;
  while (k<=N)
      {if ((a[k]==x) && (z>0))
          {f=k; z=mysteryfunction(z);}
          k=k+1; }
  ```

- **We want to reason about this program without having to analyze function mysteryfunction().**

  $$P = \{(s, s')| a[f'] = x \land 1 \leq f' \leq N \land a' = a \land x' = x\}.$$
Relative Correctness for (possibly) Non Deterministic Programs:

- $P'$ is more-correct than $P$ with respect to $R$ iff:

\[
(R \cap P')_L \supseteq (R \cap P)_L \land (R \cap P)_L \cap \overline{R} \cap P' \subseteq P
\]

- $P'$ has a larger competence domain, and on the competence domain of $P$, whenever $P'$ violates $R$, so does $P$.
- $P'$ obeys $R$ more often and violates $R$ less egregiously (in fewer ways) than does $P$. 

Relational Mathematics for Relative Correctness
Relational Mathematics for Relative Correctness

Graphic Representation:

Figure 8: $P' \sqsupseteq_R P$, Non-deterministic Specifications
Relational Mathematics for Relative Correctness

P:  int k; int z; z=1; k=1;
   while (k<=N)
      {if ((a[k]==x) && (z>0)) {f=k; z=mysteryfunction(z);}
       k=k+1;}

P': int k; int z; z=1; k=1;
    while (a[k]!=x) {k=k+1;} f=k; k=k+1;
    while (k<=N)
      {if ((a[k]==x) && (z>0)) {f=k; z=mysteryfunction(z);}
       k=k+1;}

P": int k; int z; z=1; k=1;
    while (k<=N)
      {if ((a[k]==x) && (z+5>0)) {f=k; z=mysteryfunction(z);}
       k=k+1;}
Relational Mathematics for Relative Correctness

Illustration:
- \( P' \) is more-correct than \( P \) with respect to specification \( R \): It generates one fewer incorrect output:
  - If \( x \) occurs in \( n \) locations in \( a \), say \( f_1 \ldots f_n \), then \( P \) may return any of these, including the first \( n-1 \), which are all incorrect.
  - Whereas \( P' \) may only return \( f_2 \) to \( f_n-1 \); hence it violates \( R \) in fewer ways.
- \( P'' \) is more reliable than \( P \) with respect to any probability density over \( \text{dom}(R) \), but it not more-correct than \( P \) with respect to \( R \).
  - Page 21 applies for deterministic programs, not for non-deterministic programs.
  - Reliability: a stochastic attribute; deals with plausibility.
  - Relative correctness: a logical attribute; deals with possibility.
Relative Correctness Operator: The Projection.

■ Given a Specification $R$,

\[ R = \{ (s, s') | x' = x + y \} \]

■ A Program $P$,

\[ \{ \text{while } (y! = 0) \{ x=x+1; y=y-1; \} \} \]

■ Program Function:

\[ P = \{ (s, s') | y \geq 0 \land x' = x + y \land y' = 0 \} \]

■ What functionality of $P$ is mandated by $R$?
Relational Mathematics for Relative Correctness

Relative Correctness Operator: The Projection.

- Given a Specification $R$,
  \[ R = \{(s, s') | x' = x + y\} \]

- A Program $P$,
  \[ \text{while } (y! = 0) \{ x=x+1; \ y=y-1; \} \]

- Program Function:
  \[ P = \{(s, s') | y \geq 0 \land x' = x + y \land y' = 0\} \]

- What functionality of $P$ is mandated by $R$?
  \[ \Pi_R(P) = \{(s, s') | y \geq 0 \land x' = x + y\} \]
Relational Mathematics for Relative Correctness

\[ R \]
\[ \phi_R(P) \]
\[ \Pi_R(P) \]
\[ \xi_R(P) \]
\[ P \]
Relational Mathematics for Relative Correctness

- **Projection of P over R**: What R mandates and P delivers.
  \[ \Pi_R(P) = (R \cap P) \setminus (R \cup P). \]

- **Functional Deficit of P wrt R**: What R mandates but P fails to deliver:
  \[ \phi_R(P) = R \cup \Pi_R(P). \]

- **Functional Excess of P wrt R**: What P delivers but R does not require:
  \[ \xi_R(P) = P \cup \Pi_R(P). \]
Relational Mathematics for Relative Correctness

\[ R = \{(s, s') | x' = x + y\}\]

\[ P = \{(s, s') | y \geq 0 \land x' = x + y \land y' = 0\}\]

- **Projection of \( P \) over \( R \):** What \( R \) mandates and \( P \) delivers.
  \[ \Pi_R(P) = \{(s, s') | y \geq 0 \land x' = x + y\}. \]

- **Functional Deficit of \( P \) wrt \( R \):** What \( R \) mandates but \( P \) fails to deliver:
  \[ \phi_R(P) = \{(s, s') | y < 0 \land x' = x + y\}. \]

- **Functional Excess of \( P \) wrt \( R \):** What \( P \) delivers but \( R \) does not require:
  \[ \xi_R(P) \approx \{(s, s') | y \geq 0 \land y' = 0\}. \]
Relational Mathematics for Relative Correctness

- Program $P$: Join of Projection (which $R$ mandates) and Excess (which $R$ does not mandate).

$$P \subseteq \Pi_R(P) \sqcup \xi_R(P).$$

Alternatively: what is determined by the specification vs what is determined by the design.

- Specification $R$: Join of Projection (what $P$ delivers) and Deficit (which $P$ fails to deliver).

$$R \subseteq \Pi_R(P) \sqcup \phi_R(P).$$
Relational Mathematics for Relative Correctness

Properties of the Projection (SCP 2017)

- Idempotence: Projection of the Projection is the Projection.
- All equally correct programs have the same projection.
- The projection is the least refined program in its class.
- $P'$ is more correct than $P$ with respect to $R$ if and only if: The projection of $P'$ over $R$ refines that of $P$ over $R$.
  - Redefine relative correctness.
  - Analogy: Better horseboating arrangement.
  - For $P, P'$ deterministic: $P'$ is more-correct than $P$ with respect to $R$ if and only if $P'$ refines the projection of $P$ over $R$.
- $P$ is correct with respect to $R$ if and only if the projection of $P$ over $R$ equals $R$. 
Relational Mathematics for Relative Correctness
Overview

• Premise/ Agenda
• Relational Mathematics for Absolute Correctness
• Relational Mathematics for Relative Correctness
• Correctness Enhancement: A Pervasive Paradigm
• Implications
• Conclusion
Correctness Enhancement: A Pervasive Paradigm

Program Construction: Mapping a specification into an artifact that has two attributes:

- It is correct,
- It is executable.

Stepwise Iterative Process:

- Maintain correctness, enhance executability.
  - Gold standard of program construction.
- Maintain executability, enhance correctness.
  - Refinement 2015 (Oslo).
Correctness Enhancement: A Pervasive Paradigm

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Refinement Based</th>
<th>Based on Relative Correctness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization</td>
<td>$a = R$</td>
<td>$a = \text{abort}$</td>
</tr>
<tr>
<td>Invariant Assertion</td>
<td>$a$ is correct</td>
<td>$a$ is a program</td>
</tr>
<tr>
<td>Variant Function</td>
<td>$a$ increasingly concrete (program-like)</td>
<td>$a$ increasingly correct</td>
</tr>
</tbody>
</table>
| Stepwise Validity Test | Refinement:  
$P' \sqsubseteq P$ | Rel. Correctness:  
$P' \sqsubseteq_R P$ |
| Exit test          | when $a$ is a program                                 | when $a$ is correct            |
Correctness Enhancement: A Pervasive Paradigm

Correctness Preserving vs. Correctness Enhancing Transformations
Correctness Enhancement: A Pervasive Paradigm

Question: Which is better?

- Fortunately, it does not matter.
  - Foolishness: challenging 40 years of programming wisdom.
- We argue for correctness enhancement
  - Not because it is better than correctness preservation at program derivation.
  - But rather: Unlike correctness preservation, correctness enhancement models not only program derivation, but many other processes of software engineering.
Correctness Enhancement: A Pervasive Paradigm

Corrective Maintenance

- Removing Faults (RAMICS 2014, 2015):
  - What is a fault? What is a Fault Removal?
Correctness Enhancement: A Pervasive Paradigm

Corrective Maintenance

- Removing Faults (RAMICS 2014, 2015):
  - What is a fault? What is a Fault Removal?
  - Fault: a feature \( f \) in a program \( P \) that admits a substitute \( f' \) that would make the program more-correct with respect to \( R \).
  - Fault Removal: The pair \( (f, f') \).
Correctness Enhancement: A Pervasive Paradigm

Implications of this definition:

- The difference between program faults and bad apples.
  - If we have $N$ bad apples and we remove one, we are left with $N-1$.
  - But if we have $N$ faults and we remove one, the number of remaining faults is indetermined.
- Measuring Faultiness: \textit{Fault depth}.
  - Number of (elementary) Fault Removals to a correct program.
Correctness Enhancement: A Pervasive Paradigm

Implications of this definition:

- Debugging Without Testing (ICST 2016, Chicago).
  - Four steps in debugging:
    1. Evidence of existence of a Fault.
    2. Localization of the Fault.
    3. Indication on how to fix the Fault.
    4. Evidence of fault removal.
  - All without testing; rather by static analysis.
Correctness Enhancement: A Pervasive Paradigm

Program Repair (formaliSE 2017, Buenos Aires)

- Much work on program repair, since about 2010.
- Many tools available online.
- Yet: No definition of relative correctness.
  - Correctness $\rightarrow$ Program Construction.
  - Relative Correctness $\rightarrow$ Program Repair.
Correctness Enhancement: A Pervasive Paradigm

Absence of Relative Correctness $\rightarrow$ Approximations stemming from absolute correctness.

- Sufficient (unnecessary) conditions of correctness
  - $\rightarrow$ Loss of recall.

- Necessary (non-sufficient) conditions of correctness
  - $\rightarrow$ Loss of precision.

- Failure to recognize scale of elementary fault:
  - Failure to distinguish between single two-site fault and two single-site faults.
  - Unwarranted Combinatorial explosion.

- Failure to recognize stepwise Improvements in RC
  - Let the Program Expose its Faults in the order it chooses.
Correctness Enhancement: A Pervasive Paradigm

Algorithm for Program Repair:

- Inputs: Program $P$, specification $R(s,s')$, predicate $domR(s)$, test data $T$.
- Output: Program $P'$, absolutely correct with respect to $T \setminus R$.
  - More-correct than $P$ with respect to $R$.

Loop:

\[ \{ P' = P; \text{ while not abs cor}(P') \{ P' = \text{enhance-correctness}(P); \} \} \]

Precise Oracles using $R()$ and $domR()$ for:

- Absolute Correctness.
- Relative Correctness.
- Strict Relative Correctness.
Correctness Enhancer: A Pervasive Paradigm

- Six modifications (not faults)
  - Only one fault is visible.
  - Faults mask each other.
  - Fault Density = 1.
  - Fault depth = 5.
- Depth decreases by 1:
  - $\text{depth}(P') = \text{depth}(P) - 1$.
- Density does not:
  - $\text{density}(P) = 1$.
  - $\text{density}(M79) = 3$. 
Correctness Enhancement: A Pervasive Paradigm

Program Merger

- Specification $R$,
- Two programs, $P1$ and $P2$.
  - Both incorrect,
  - Each covering part of $R$’s requirements,
  - Merger: $P'$ that merges the relevant functionality of $P1$ and $P2$. 
Correctness Enhancement: A Pervasive Paradigm

Requirements on $P'$:

- $P'$ refines $P_1$ and refines $P_2$? No, for two reasons:
  - Refining $P_1$ and $P_2$ may be impossible.
    - Incompatibility between functional excesses.
  - Refining $P_1$ and $P_2$ is unnecessary.
    - Sufficient: $P'$ more correct than $P_1$ and $P_2$.
    - Equivalent: $P'$ refines projections of $P_1$ and $P_2$.
    - Good news: projections are compatible, admit a join.
Correctness Enhancement: A Pervasive Paradigm

(a) Merger of $P_1$ and $P_2$
Correctness Enhancement: A Pervasive Paradigm

Program Upgrade

- Specification $R$,
- Sprawling Application $P$.
  - E.g. Enterprise DP application
- New requirement $Q$.
  - New functionality we want to add to $P$.
  - E.g. New input screen, new report, new function to compute.
Correctness Enhancement: A Pervasive Paradigm

Requirement on $P'$ to be an upgrade of $P$ with $Q$?

- $P'$ must refine $P$ and $Q$.
  - Not necessary; and may be impossible.
    - $Q$ incompatible with functional excess of $P$.
- $P'$ must refine $R$ and $Q$.
  - No assurance that $P$ refines $R$; why expect $P'$?
- $P'$ must refine $Q$ and be more-correct than $P$ with respect to $R$.
  - In other words, satisfy $Q$ without degrading $P$.
  - Possible if $Q$ and projection of $P$ are compatible.
Correctness Enhancement: A Pervasive Paradigm

(b)
Upgrading $P_1$ with Feature $Q$
Overview

• Premise/ Agenda
• Relational Mathematics for Absolute Correctness
• Relational Mathematics for Relative Correctness
• Correctness Enhancement: A Pervasive Paradigm
• Implications
• Conclusion
Implications

- Sound scientific basis for engineering
- Explicit criteria for successful completion.
- Modeling heretofore informal/vague processes.
- Efficient/goal-oriented tasks.
- Defining Faults
  - Debugging without testing.
  - Fault Density vs Fault Depth.
  - Program Repair under combinatorial control.
  - Removing Faults at $7.99 a piece or less.
Overview

• Premise/ Agenda
• Relational Mathematics for Absolute Correctness
• Relational Mathematics for Relative Correctness
• Correctness Enhancement: A Pervasive Paradigm
• Implications
• Conclusion
Conclusion

(a) Programming for Correctness

(b) Programming for Reliability

(c) Adaptive Maintenance

(d) Corrective Maintenance
Conclusion

(a) Program Merger

(b) Program Upgrade
Conclusion

Observation

- Mr Jourdain: Surprised to find that he has been saying prose for 40 years and did not know it.
- Software engineers entitled to an even bigger surprise: we have been practicing correctness enhancement for 50 years and many (perhaps most?) of us did not know it.

In order for this discovery to do us any good, we must:
- Expand our understanding of relative correctness/ correctness enhancement.
- Expand our understanding of how and to what extent it pervades software engineering.
- Consider how this may be used to improve software engineering practice.
Conclusion

Research Agenda

- From verifying relative correctness to generating more-correct-by-design artifacts
  - As was done in the eighties and nineties for absolute correctness (Hoare, Dijkstra, Gries, Hehner, Morgan).
  - Significant hurdle: relative correctness is context-sensitive.
    - Scaling specifications down to artifacts.
  - Significant payoff: commensurate with pervasiveness.
Conclusion

Research Agenda

- From verifying relative correctness to generating more-correct-by-design artifacts
  - As was done in the eighties and nineties for absolute correctness (Hoare, Dijkstra, Gries, Hehner, Morgan).
  - Significant hurdle: relative correctness is context-sensitive.
    - Scaling specifications down to artifacts.
    - Significant payoff: commensurate with pervasiveness.

- Other Forms of Refinement.
Overview

• Premise/ Agenda
• Relational Mathematics for Absolute Correctness
• Relational Mathematics for Relative Correctness
• Correctness Enhancement: A Pervasive Paradigm
• Implications
• Conclusion