Formal verification of economic mechanisms

Justin Hsu (PI), Joseph Halpern (co-PI), **Noah Bertram** (PhD student), Cornell University

Project Aim

Develop automated techniques for verifying various properties of economic mechanisms

What do we mean by economic mechanisms?

Economic mechanisms are procedures that allocate various kinds of goods to a set of agents, which are algorithmic in style, and may involve participation of the agents.

Examples:

- Good exchange
- Agent matching
- Good allocation
- Auctions

Types of properties to verify

Basic well-formedness

Incentive compatability

Fairness guarantees

Efficiency guarantees

Verification challenges

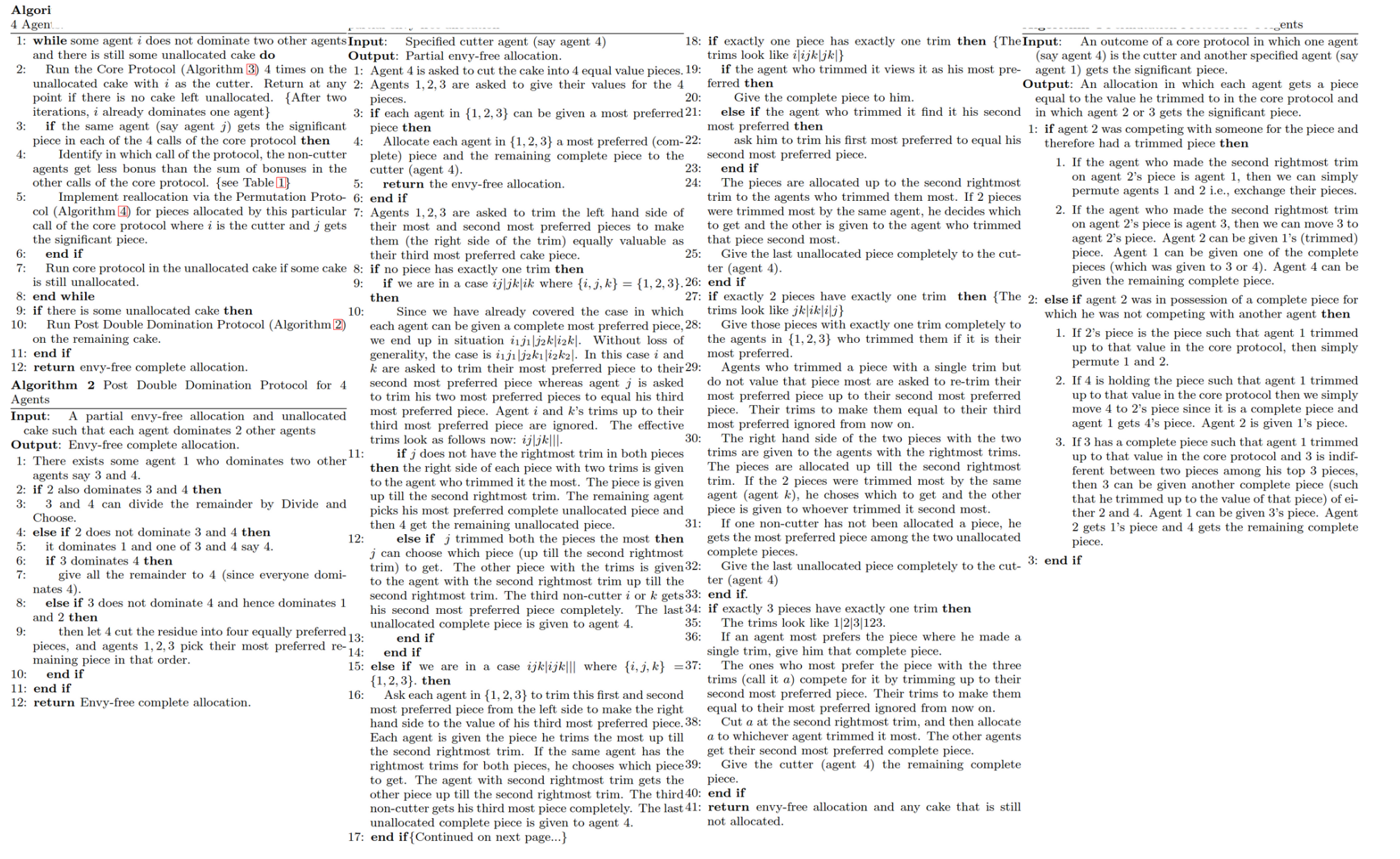
Each problem setting requires its own solution

Typically no pre-existing formal semantics

Requires reasoning about both agent preferences and behavior

Mechanisms are often complex and difficult to describe

Example cake-cutting protocol:
Envy-free cake-cutting protocol for 4 agents



```
Algori
4 Agen
while some agent $i$ does not dominate two other agents and there is still some unallocated cake do
    Run the Core Protocol (Algorithm 3) 4 times on the unallocated cake with $i$ as the cutter. Return at any point if there is no cake left unallocated. {After two iterations, $i$ already dominates one agent}
    if the same agent (say agent $j$) gets the significant piece in each of the 4 calls of the core protocol then
        Identify in which call of the protocol, the non-cutter agents get less bonus than the sum of bonuses in the other calls of the core protocol. {see Table 1}
        Implement reallocation via the Permutation Protocol (Algorithm 4) for pieces allocated by this particular call of the core protocol where $i$ is the cutter and $j$ gets the significant piece.
    end if
    Run core protocol in the unallocated cake if some cake is still unallocated.
end while
if there is some unallocated cake then
    Run Post Double Domination Protocol (Algorithm 2) on the remaining cake.
end if
return envy-free complete allocation.

Algorithm 2 Post Double Domination Protocol for 4 Agents
Input: A partial envy-free allocation and unallocated cake such that each agent dominates 2 other agents
Output: Envy-free complete allocation.
There exists some agent 1 who dominates two other agents say 3 and 4.
if 2 also dominates 3 and 4 then
    3 and 4 can divide the remainder by Divide and Choose.
else if 2 does not dominate 3 and 4 then
    it dominates 1 and one of 3 and 4 say 4.
    if 3 dominates 4 then
        give all the remainder to 4 (since everyone dominates 4).
    else if 3 does not dominate 4 and hence dominates 1 and 2 then
        then let 4 cut the residue into four equally preferred pieces, and agents 1, 2, 3 pick their most preferred remaining piece in that order.
    end if
end if
return Envy-free complete allocation.
```

```
Input: Specified cutter agent (say agent 4)
Output: Partial envy-free allocation.
Agent 4 is asked to cut the cake into 4 equal value pieces.
Agents 1, 2, 3 are asked to give their values for the 4 pieces.
if each agent in $\{1,2,3\}$ can be given a most preferred piece then
    Allocate each agent in $\{1,2,3\}$ a most preferred (complete) piece and the remaining complete piece to the cutter (agent 4).
    return the envy-free allocation.
end if
Agents 1, 2, 3 are asked to trim the left hand side of their most and second most preferred pieces to make them (the right side of the trim) equally valuable as their third most preferred cake piece.
if no piece has exactly one trim then
    if we are in a case $ij|jk|ik$ where $\{i,j,k\}=\{1,2,3\}$ then
        Since we have already covered the case in which each agent can be given a complete most preferred piece, we end up in situation $i_1j_1|j_2k|i_2k|$. Without loss of generality, the case is $i_1j_1|j_2k_1|i_2k_2|$. In this case $i$ and $k$ are asked to trim their most preferred piece to their second most preferred piece whereas agent $j$ is asked to trim his two most preferred pieces to equal his third most preferred piece. Agent $i$ and $k$'s trims up to their third most preferred piece are ignored. The effective trims look as follows now: $ij|jk|||$.
        if $j$ does not have the rightmost trim in both pieces then the right side of each piece with two trims is given to the agent who trimmed it the most. The piece is given up till the second rightmost trim. The remaining agent picks his most preferred complete unallocated piece and then 4 get the remaining unallocated piece.
        else if $j$ trimmed both the pieces the most then $j$ can choose which piece (up till the second rightmost trim) to get. The other piece with the trims is given to the agent with the second rightmost trim up till the second rightmost trim. The third non-cutter $i$ or $k$ gets his second most preferred piece completely. The last unallocated complete piece is given to agent 4.
        end if
    end if
else if we are in a case $ijk|ijk|||$ where $\{i,j,k\}=\{1,2,3\}$. then
    Ask each agent in $\{1,2,3\}$ to trim this first and second most preferred piece from the left side to make the right hand side to the value of his third most preferred piece. Each agent is given the piece he trims the most up till the second rightmost trim. If the same agent has the rightmost trims for both pieces, he chooses which piece to get. The agent with second rightmost trim gets the other piece up till the second rightmost trim. The third non-cutter gets his third most piece completely. The last unallocated complete piece is given to agent 4.
end if{Continued on next page...}
if exactly one piece has exactly one trim then {The trims look like $i|ijk|jk|$}
    if the agent who trimmed it views it as his most preferred then
        Give the complete piece to him.
    else if the agent who trimmed it find it his second most preferred then
        ask him to trim his first most preferred to equal his second most preferred piece.
    end if
    The pieces are allocated up to the second rightmost trim to the agents who trimmed them most. If 2 pieces were trimmed most by the same agent, he decides which to get and the other is given to the agent who trimmed that piece second most.
    Give the last unallocated piece completely to the cutter (agent 4).
end if
if exactly 2 pieces have exactly one trim then {The trims look like $jk|ik|i|j$}
    Give those pieces with exactly one trim completely to the agents in $\{1,2,3\}$ who trimmed them if it is their most preferred.
    Agents who trimmed a piece with a single trim but do not value that piece most are asked to re-trim their most preferred piece up to their second most preferred piece. Their trims to make them equal to their third most preferred ignored from now on.
    The right hand side of the two pieces with the two trims are given to the agents with the rightmost trims. The pieces are allocated up till the second rightmost trim. If the 2 pieces were trimmed most by the same agent (agent $k$), he choses which to get and the other piece is given to whoever trimmed it second most.
    If one non-cutter has not been allocated a piece, he gets the most preferred piece among the two unallocated complete pieces.
    Give the last unallocated piece completely to the cutter (agent 4)
end if.
if exactly 3 pieces have exactly one trim then
    The trims look like 1|2|3|123.
    If an agent most prefers the piece where he made a single trim, give him that complete piece.
    The ones who most prefer the piece with the three trims (call it $a$) compete for it by trimming up to their second most preferred piece. Their trims to make them equal to their most preferred ignored from now on.
    Cut $a$ at the second rightmost trim, and then allocate $a$ to whichever agent trimmed it most. The other agents get their second most preferred complete piece.
    Give the cutter (agent 4) the remaining complete piece.
end if
return envy-free allocation and any cake that is still not allocated.
```

```
ents
Input: An outcome of a core protocol in which one agent (say agent 4) is the cutter and another specified agent (say agent 1) gets the significant piece.
Output: An allocation in which each agent gets a piece equal to the value he trimmed to in the core protocol and in which agent 2 or 3 gets the significant piece.
if agent 2 was competing with someone for the piece and therefore had a trimmed piece then
    1. If the agent who made the second rightmost trim on agent 2's piece is agent 1, then we can simply permute agents 1 and 2 i.e., exchange their pieces.
    2. If the agent who made the second rightmost trim on agent 2's piece is agent 3, then we can move 3 to agent 2's piece. Agent 2 can be given 1's (trimmed) piece. Agent 1 can be given one of the complete pieces (which was given to 3 or 4). Agent 4 can be given the remaining complete piece.
else if agent 2 was in possession of a complete piece for which he was not competing with another agent then
    1. If 2's piece is the piece such that agent 1 trimmed up to that value in the core protocol, then simply permute 1 and 2.
    2. If 4 is holding the piece such that agent 1 trimmed up to that value in the core protocol then we simply move 4 to 2's piece since it is a complete piece and agent 1 gets 4's piece. Agent 2 is given 1's piece.
    3. If 3 has a complete piece such that agent 1 trimmed up to that value in the core protocol and 3 is indifferent between two pieces among his top 3 pieces, then 3 can be given another complete piece (such that he trimmed up to the value of that piece) of either 2 and 4. Agent 1 can be given 3's piece. Agent 2 gets 1's piece and 4 gets the remaining complete piece.
end if
```

H Aziz, S Mackenzie (2015)

Scientific Impact...

of the project as a whole:

Formalization of mechanisms as programs

Characterize mechanism correctness proof structure

Formal assurance of mechanism correctness

of Slice so far:

Developed formal semantics for cake-cutting protocols

Reduced cake-cutting protocol correctness to linear real arithmetic

Formally verified envy-freeness of many protocols, including one for four agents

Slice: a language for cake-cutting

Found in PLDI'23 and CAV'24 proceedings

What is cake-cutting?

Cake-cutting aims to divide an infinitely divisible good among a set of agents in a fair way. In this setting, agents have measure-like preferences over the cake, and are not in general the same preferences for all agents.

Example: Cut-Choose

In this two agent protocol, one agent cuts the cake into two equally preferred pieces, while the other agent takes their preferred piece.

```
let $m$ = mark$_1$(cake, $1/2$) in
let $i_1, i_2$ = divide(cake, $m$) in
if eval$_2(i_1) \geq$ eval$_2(i_2)$ then
    $(i_2, i_1)$
else
    $(i_1, i_2)$
```

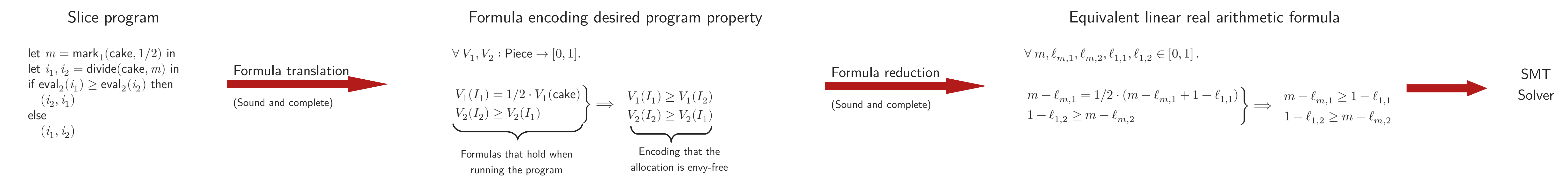
Cut-Choose is **envy-free**:

Neither agent prefers what the other received to what they received.

Slice verification results

Protocol	Program size (lines)	Formula size (lines)	SMT solving time (s)
Cut-Choose	6	35	0.00
Surplus	11	56	0.00
Waste-Makes-Haste-3	8	924	0.02
Selfridge-Conway-Surplus	19	7726	0.01
Selfridge-Conway-Full	21	98292	0.46
Aziz-Mackenzie-3	23	8086180	6.82
Waste-Makes-Haste-4	290	157553237	82

Slice verification pipeline



Slice program

```
let $m$ = mark$_1$(cake, $1/2$) in
let $i_1, i_2$ = divide(cake, $m$) in
if eval$_2(i_1) \geq$ eval$_2(i_2)$ then
    $(i_2, i_1)$
else
    $(i_1, i_2)$
```

Formula translation

(Sound and complete)

Formula encoding desired program property

$$\forall V_1, V_2 : \text{Piece} \to [0,1].$$

$$\underbrace{\left.\begin{array}{l} V_1(I_1) = 1/2 \cdot V_1(\text{cake}) \\ V_2(I_2) \geq V_2(I_1) \end{array}\right\}}_{\text{Formulas that hold when running the program}} \implies \underbrace{\begin{array}{l} V_1(I_1) \geq V_1(I_2) \\ V_2(I_2) \geq V_2(I_1) \end{array}}_{\text{Encoding that the allocation is envy-free}}$$

Formula reduction

(Sound and complete)

Equivalent linear real arithmetic formula

$$\forall m, \ell_{m,1}, \ell_{m,2}, \ell_{1,1}, \ell_{1,2} \in [0,1].$$

$$\left.\begin{array}{l} m - \ell_{m,1} = 1/2 \cdot (m - \ell_{m,1} + 1 - \ell_{1,1}) \\ 1 - \ell_{1,2} \geq m - \ell_{m,2} \end{array}\right\} \implies \begin{array}{l} m - \ell_{m,1} \geq 1 - \ell_{1,1} \\ 1 - \ell_{1,2} \geq m - \ell_{m,2} \end{array}$$

SMT Solver

Broader Impacts

This project aims to:

Increase trust in economic mechanism correctness or find errors

Produce tools for developing formally verified implementations of mechanisms

Several economic mechanisms are used in practice!

Some examples:

- Kidney exchange
- Medical resident matching
- Telecommunication spectrum auctions

We are also starting to discuss applications of Slice with fair division researchers in Cornell ORIE and elsewhere.

This project has supported:

- A PhD student
- Two undergradute research projects

And hopefully more students to come!

Award number 2319186

The NSF Formal Methods in the Field PI Meeting (2024 FMitF PI Meeting)
November 12-13, 2024 | The University of Iowa | Iowa City, Iowa