School of Engineering | MIT Schwarzman College of Computing

MTL MICROSYSTEMS TECHNOLOGY LABORATORIES

LLM Agent Assisted Workflow for Analog Circuit Design Data Collection

Yan Xu[1], Wenjie Lu[2], Tao Yu[2], Ruonan Han[1]

[1]Department of EECS, MIT

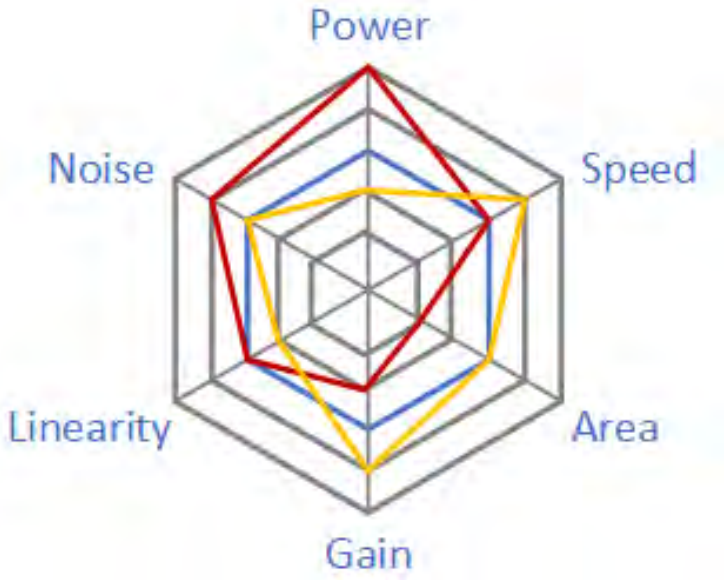
[2]Analog Device Inc.

https://hangroup.mit.edu

MIT AI Hardware Program

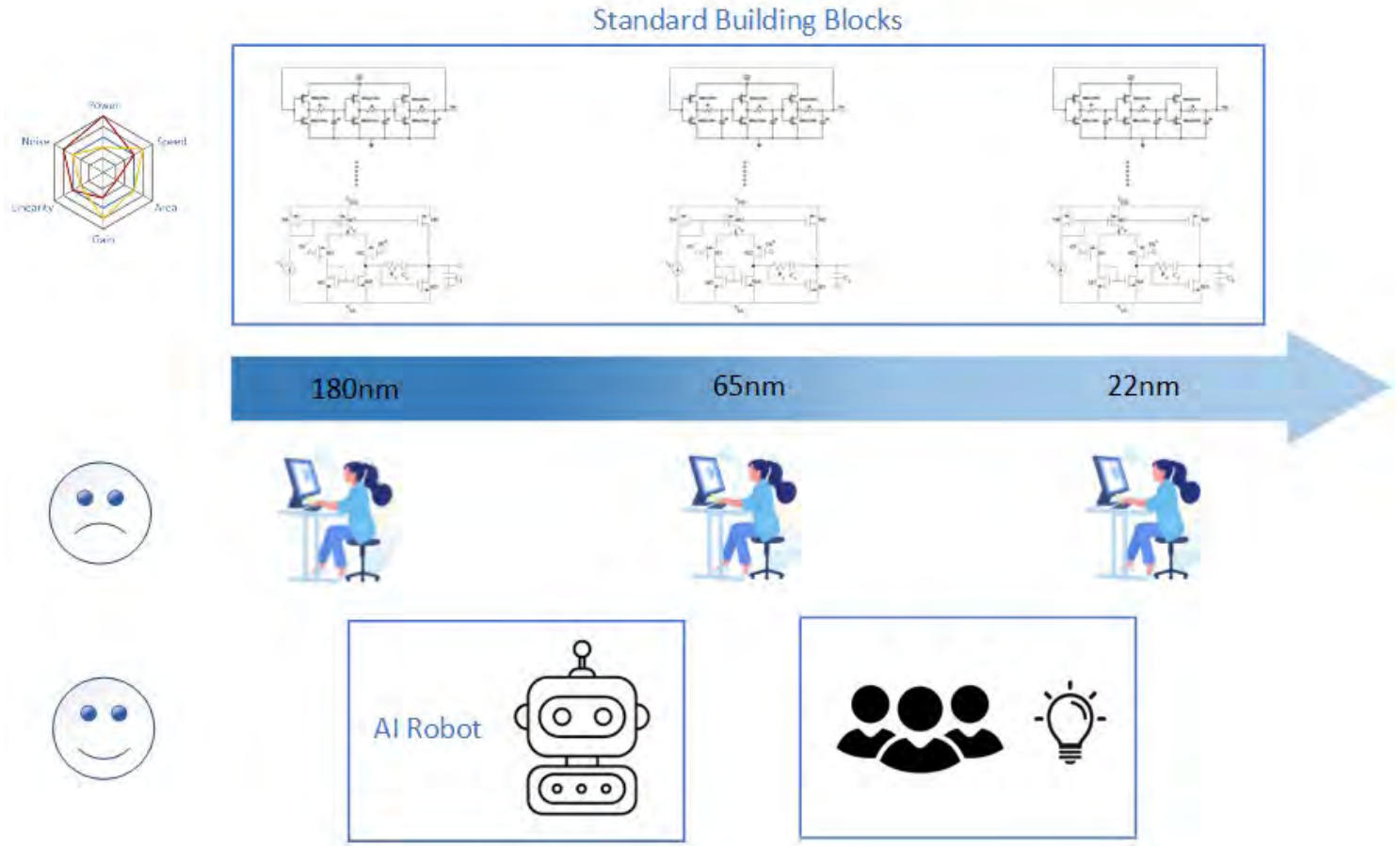
Motivation

- **Analog Circuit Automation Design Challenge**



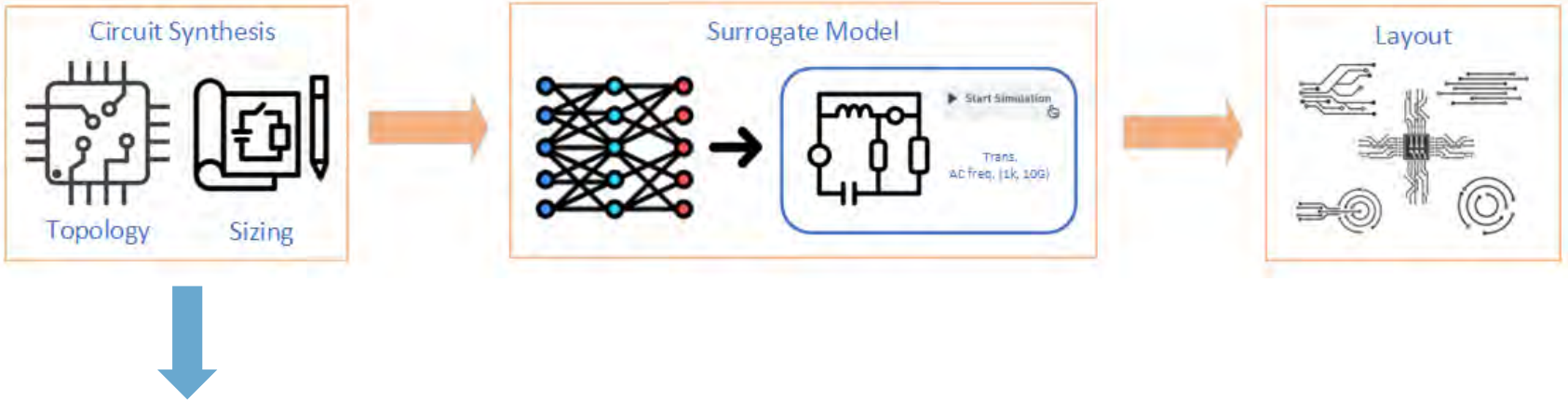
- Human Intuition
- Complex trade-off space

Vision



- AI for standard circuits design
- Human on innovative design

Problem Scope



- **Circuit Design Task refers to:**
  1. Topology Selection or discovery
  2. Parameter Sizing
- **Design Automation Goal**

Given a set of required specifications → Research system scope → Output circuit topology with parameter size

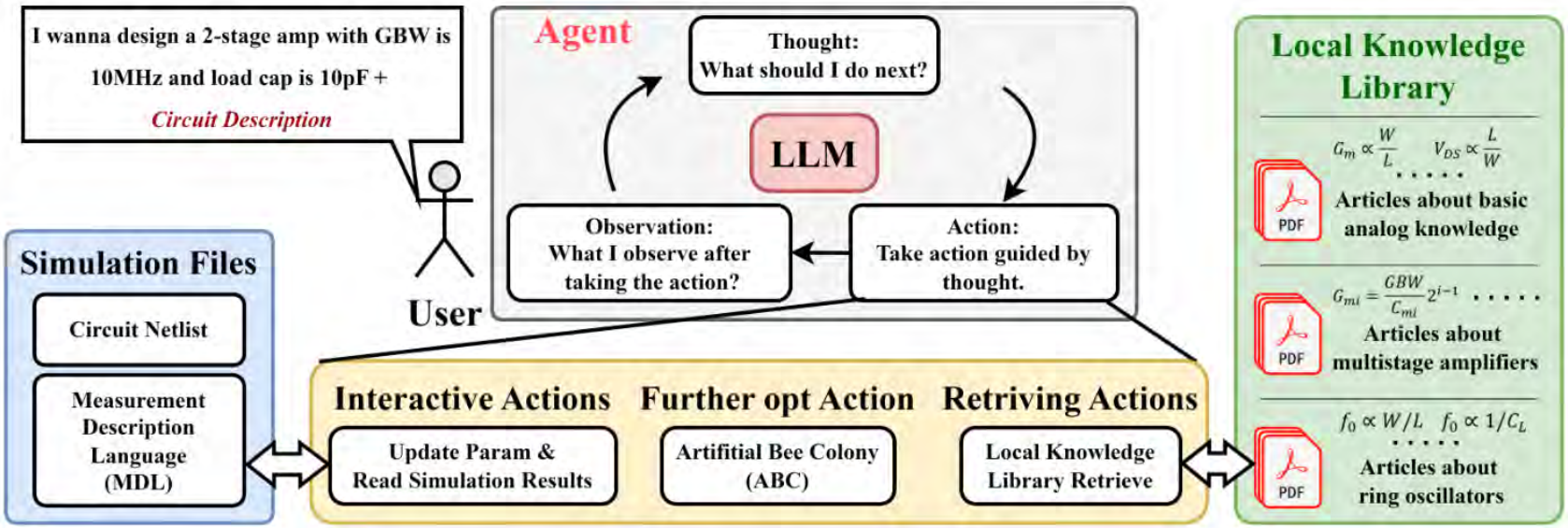
Approach: Autonomous LLM Agents

- **What Do LLM Agents Typically Mean?**
  - Given a complex task, they self-decompose, make decisions, and solve problems without human intervention

- **Can LLM Agents Magically Design Everything for Us?**
  - Hypothesis: Due to the lack of domain-specific data in LLMs, they cannot magically complete complex design tasks on their own
  - But multiple studies have been made to enable LLM agents for circuit design

Prior Work

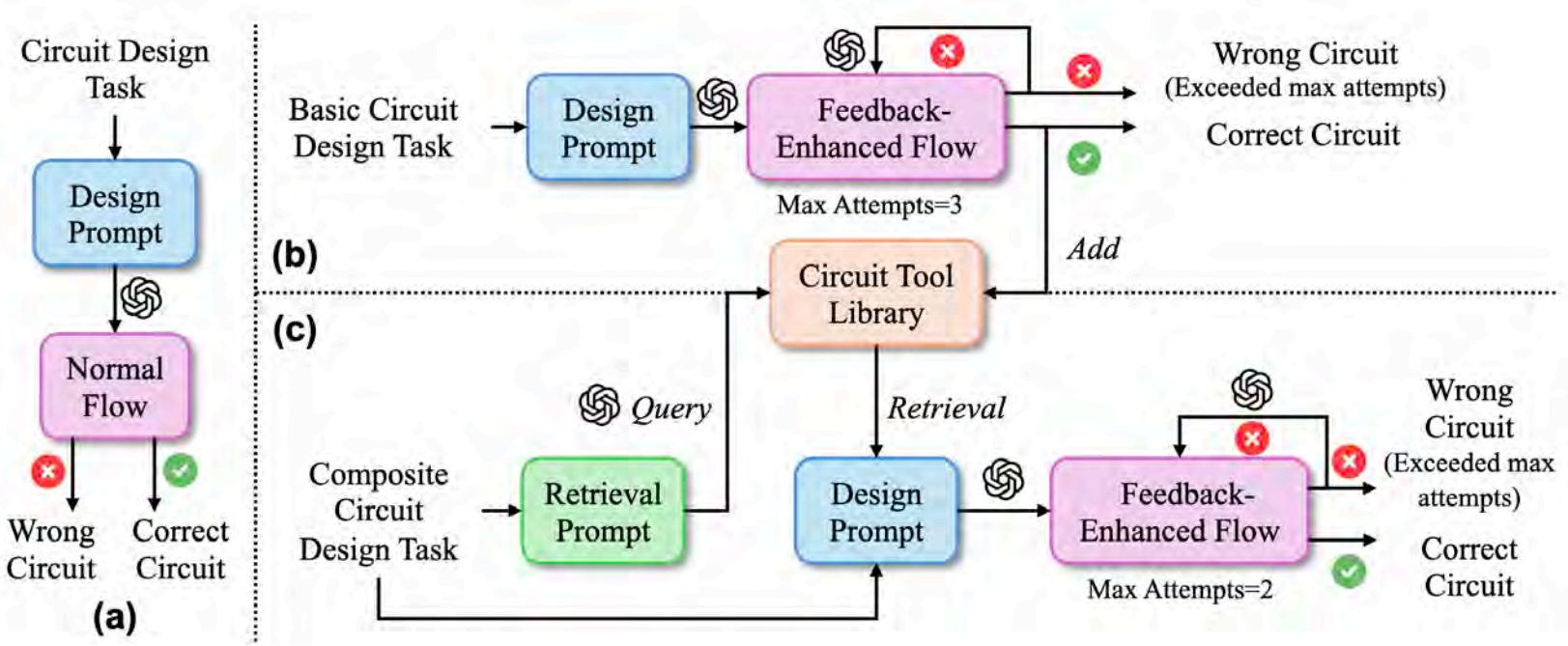
- **Domain-specified LLM agent**
  - Formulates the task as a **knowledge-based decision-making** problem
  - Solve the parameter sizing with **fully specified topology**
  - End-to-end design verification, demonstrated on three complex yet standard design cases
  - Hand-crafted knowledge bases and customized guidance for each design case are hard to scale



C. Liu, Y. Liu, et. al, “LADAC: Large Language Model-driven Auto-Designer for Analog Circuits.” TechRxiv.(2024).

Prior Work

- **AnalogCoder**
  - Represent circuit as python code, framing it as **code generation** problem
  - Focus on hierarchical topology generation guided by instructions and feedback correction **without parameter sizing**
  - Hand crafted design check rules for specific design category



Lai, Yao, Sungyoung Lee, Guojin Chen, Souradip Poddar, Mengkang Hu, David Z. Pan, and Ping Luo. "AnalogCoder: Analog Circuit Design via Training-Free Code Generation." arXiv, May 30, 2024. https://doi.org/10.48550/arXiv.2405.14918.

Start from Free-Running Exploration

🪜 **Manual efforts are hard to scale**

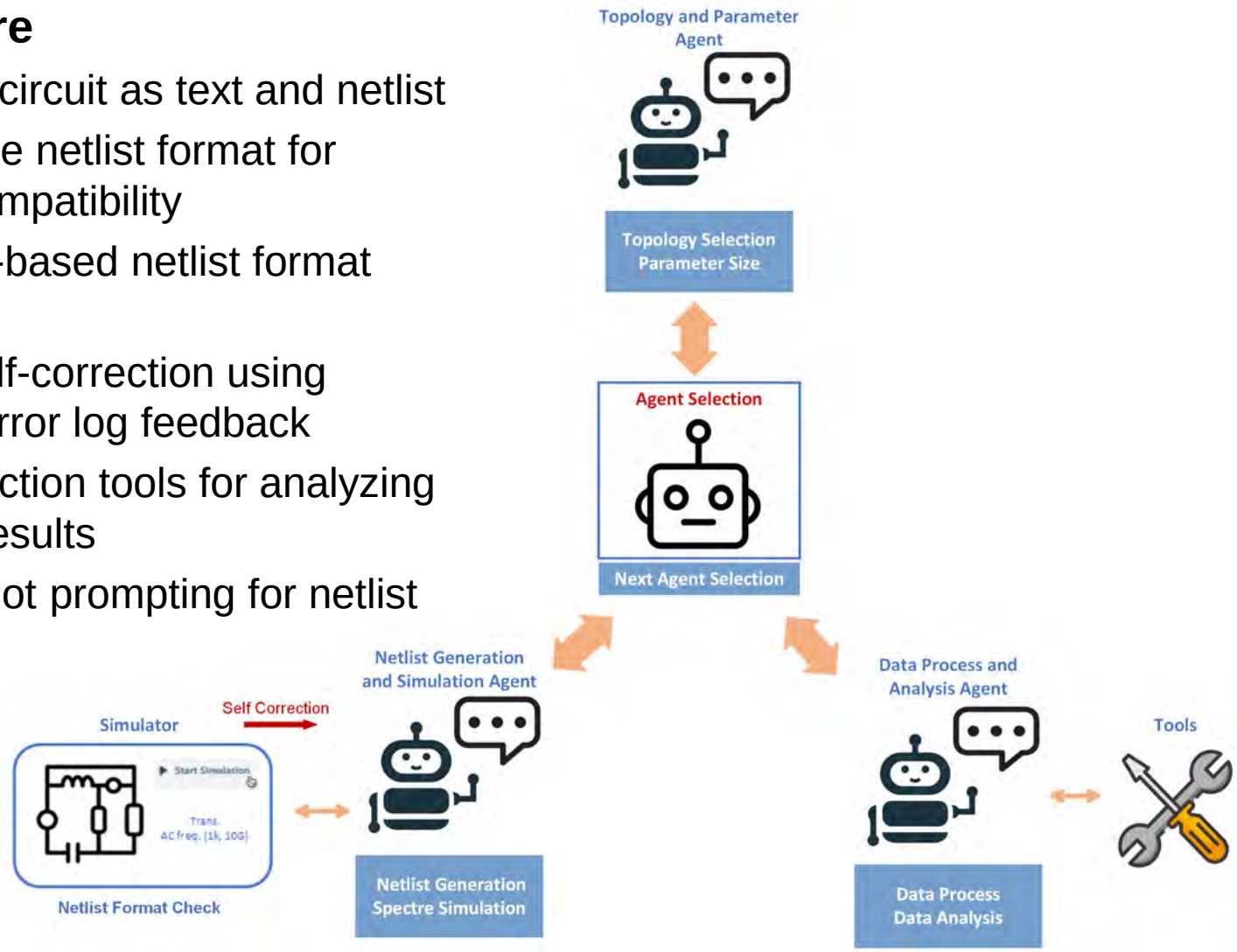
⚙️ **A standardized, method-driven workflow is needed to support diverse circuit types**

🚀 **Preliminary Experiments**

- **We begin with unconstrained, free-running agents to avoid relying on manual heuristics**
- **These serve as a baseline for comparison and to uncover limitations inherent to LLM agent**

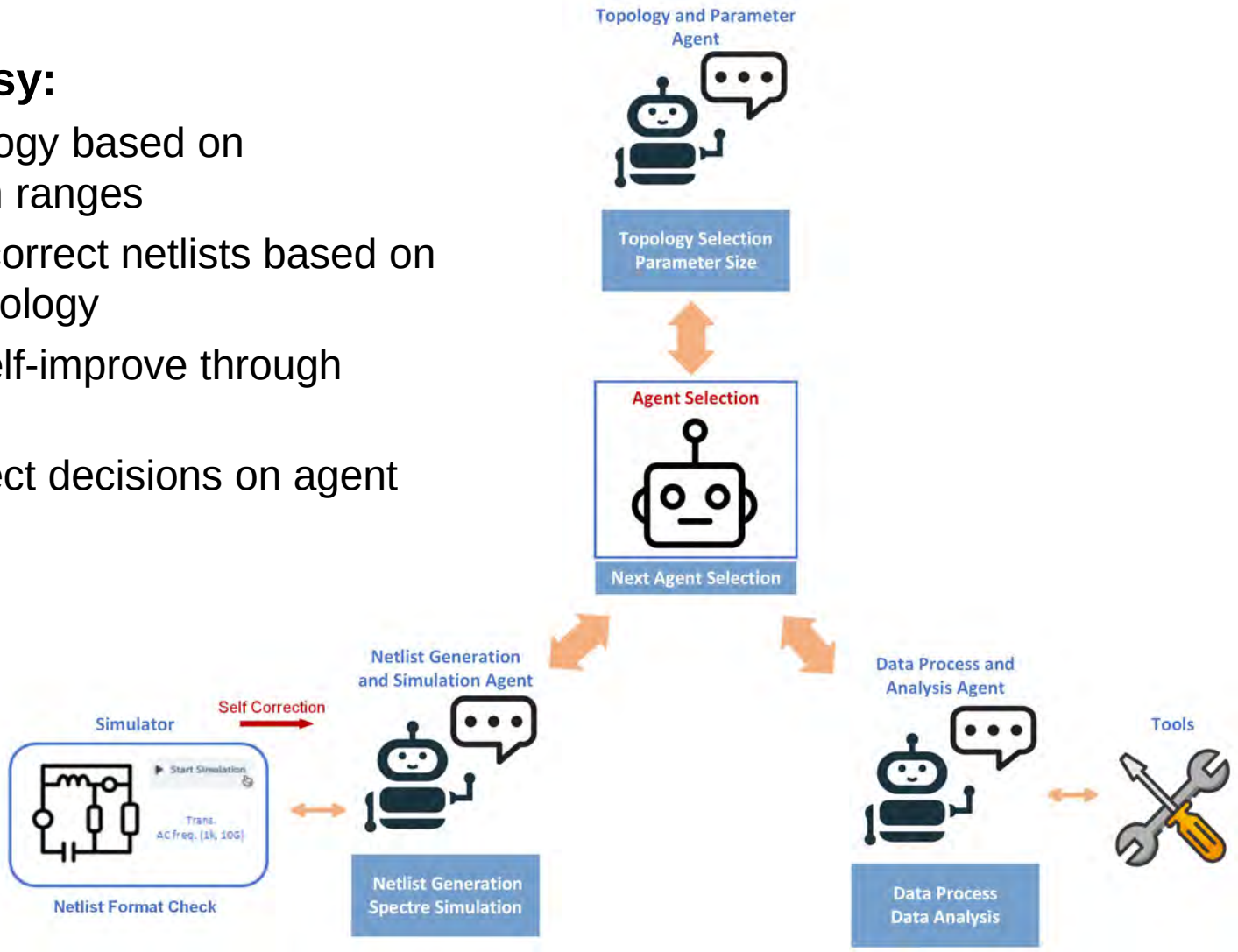
Preliminary Experiment: Multi-Agent System

- **System feature**
  - Represents circuit as text and netlist
  - Uses Spectre netlist format for Cadence compatibility
  - Applies rule-based netlist format checks
  - Supports self-correction using simulation error log feedback
  - Includes function tools for analyzing simulation results
  - Uses few-shot prompting for netlist generation



Preliminary Experiment: Multi-Agent System

- **System fantasy:**
  - Select topology based on specification ranges
  - Generates correct netlists based on selected topology
  - Learns to self-improve through feedback
  - Makes correct decisions on agent selection



LLM Multi-Agent Experiment Results

Model	Design Questions Solve Rate	Reliability
Gpt-4o-mini	1/20	5/10
O1-mini	1/20	9/10
O3-mini	1/20	9/10
Claude-3-opus	0/20	0/10
Gemini-1.5-pro	0/20	0/10

- **Only the common-source amplifier with resistive load could be successfully completed**
- **All other design cases failed due to the challenges discussed below**

LLM Multi-Agent Challenges

- **Prone to errors**
  - Netlist generation often includes **connection errors**

```
// Common source amplifier with active load
M1 (vout vin vss vss) nch w=1u l=65n
M2 (vout vb vss vss) pch w=1u l=65n
```

(can be checked by customized rule-based logic)

```
// Common source amplifier with diode-connected load
M1 (vout vin vss vss) nch w=1u l=65n
M2 (vout vout vdd vss) nch w=1u l=65n
```

(Harder to detect without context-aware validation)

- **Agent Selection Issues**
  - Gets **stuck in repetitive cycles or make incorrect selections**
  - Cycle detection helps, but it remains difficult to select the right agent to move forward
  - High communication overhead impacts both time and efficiency
- **Limited Generalization**
  - Tends to replicate training examples without new topology and sizing design

LLM Multi-Agent Challenges

- **Lacks structured reasoning for mapping high-level topology descriptions to valid netlists**
  - For example, when asked for a *common-gate amplifier,* responded with a *common-source amplifier*

User Query: Design a common gate amplifier
LLM Response: The following is common gate amplifier with resistive load M1 (vout vin vss vss) nch w=1u l=65n R1 (vout vdd) r=10k

❌ *LLM confuses circuit types due to weak topology understanding*

📝 **Without sufficient constraints or robust tool integration, the agent can only complete simple tasks with low reliability**

Training Agents: Closing the Domain Gap

🔍 **Limitations of current training-free LLMs**

- Insufficient domain-specific knowledge
- Weak domain reasoning capabilities

↓

➡ **We need to train "agents" that can embed domain knowledge and reasoning ability**

↓

📉 **Requires targeted data for “agent” training**

↓

🛠 **Solution: Workflow-guided agents to collect data**

Observations from Circuit Design Automation

Why Use Multi-Agent Systems?

- Individual LLMs lack familiarity with circuit design tasks
- Multi-agent systems compensate by dividing responsibilities

Co-Optimization is Essential

- Circuit topology and parameter sizing can be co-optimized
- This joint optimization is typically performed by **a single expert**

"Automation" = Guided Decision-Making

- A multi-agent system is essentially a decision-making pipeline
- Based on current state, agents perform sequential actions guided by prompts
- This process can be unified as **an LLM-embedded workflow** to overcome the limitations mentioned above

What Data Do We Need?

- **To address *insufficient domain-specific knowledge***

  → Requires factual, structured data on circuits, topologies, specs, and configurations

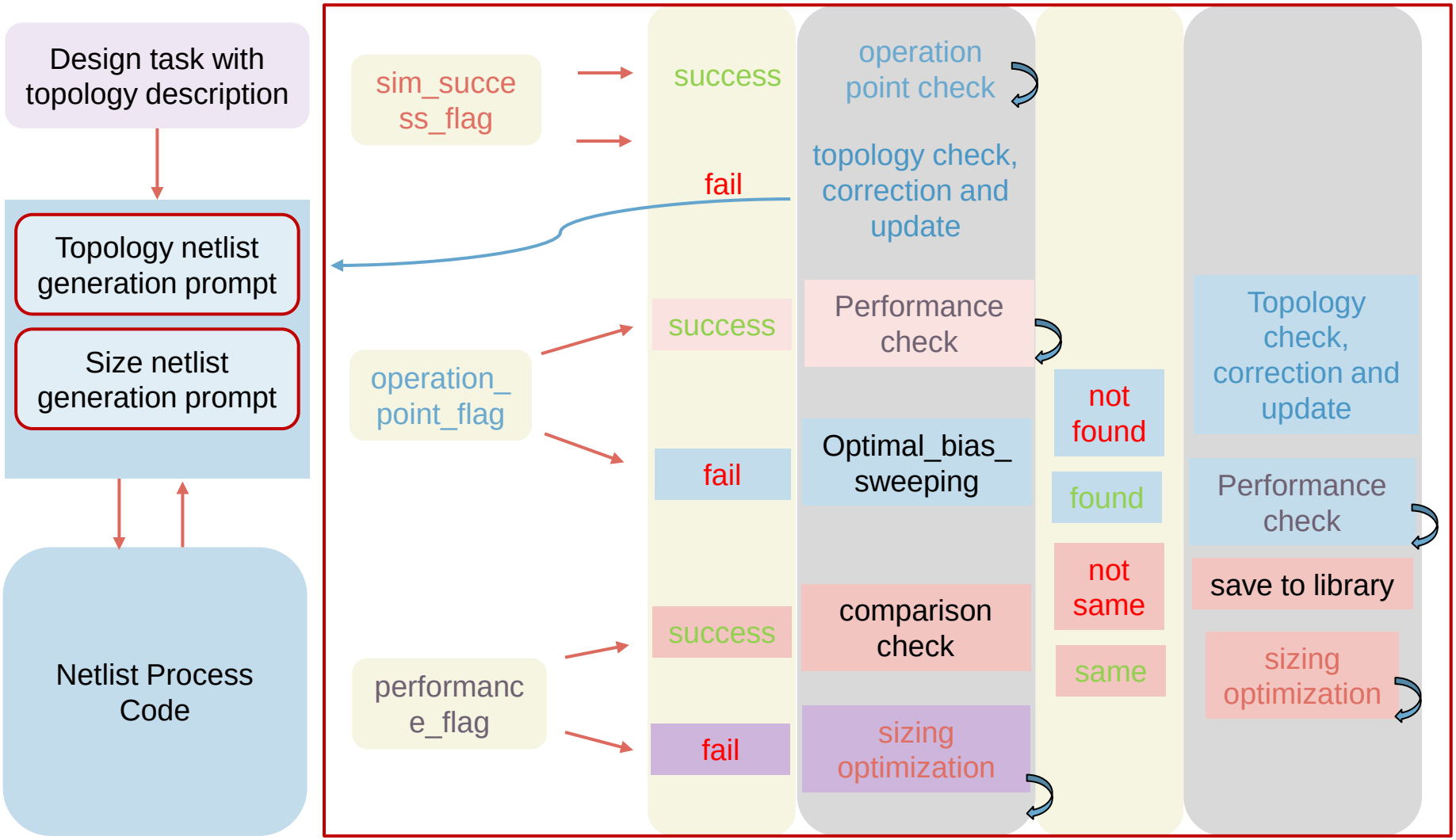
  → **Can be obtained from successfully generated design cases**

- **To address *weak domain reasoning capabilities***

  → Requires action sequences, state transitions, and design decision processes

  → **Can be extracted from the code workflow tree and its decision branches**

Workflow System: Refined Action Branching



To Improve Design Coverage and Success Rate

Key feature

- **Both topology and parameter sizing are represented as netlists**
- **Two-step generation:**
  - First generate the topology netlist, then the sizing netlist with minimal tokens generation
  - This reduces token load and the effect of limited domain knowledge
- **Standardized and scaled ways to enable generalization**

Start from simple problem settings

- **Focusing on one spec (e.g., "gain"), then scale to others using the same process**

Topology Netlist Generation

- **Constructed via Semi-Defined Function Calls**
  - Recognize that basic elements are transistors with a finite set of configurations, and passive components
  - Only generate <placeholder> including “flexible” nodes and names
  - Let the LLM select configurations and connect them to form valid topologies

Configuration	Topology Function
NMOS common source	create_nmos_common_source_input(1, <drain_connect_to>, "vin", <source_connect_to>, "vss")
NMOS diode connected load	create_nmos_diode_connected_load(1, <drain_connect_to>, <source_connect_to>, "vss")
NMOS active current source load	create_nmos_active_current_source_load_with_bias(1, <drain_connect_to>, "vb", <source_connect_to>, "vss")
....	....

Topology Netlist Generation

- **Hierarchical topology generation**

Partial Prompt:

CIRCUIT TOPOLOGY DESCRIPTION:
<Your plain text description goes here.>

```spectre
SUBCKT <NAME> vdd vss vin vout:
  <element function call lines>
  # The available element function calls:
          ....
ends <NAME>
TOP HIERARCHY SUBCKT <mask_top>
vdd vss vin vout:
ends <mask_top>
```

Partial Task description:

"topology_category": "common source amplifier with diode-connected load",
"design task":
"functional description": "voltage amplifier",
"general design specs": {
"gain/differential gain": " 2 dB",
"power": "none",
"bandwidth/speed": "none",
"noise": "none"
},

Size Netlist Generation

- **Output topology netlist (partial)**

```
M1 (vout vin vss vss) nch l=<para_length_1> w=<para_width_1>
M2 (vdd vdd vout vss) nch l= =<para_length_2> w= =<para_width_2>
```

- **Then, LLM fills in the <placeholder> for parameter sizing task**

Sizing Prompt (Partial):
PARAMETER MAPPING: <placeholder> : <assigned_value> [TYPE]

- Finally, post-processing to form the complete netlist

What data can be collected in this workflow?

Decision Making Data

- The code workflow uses multiple **flags** to mark system **states**, and each state has multiple possible **actions** to choose from
- These can be encoded as action sequences and used as training data for domain reasoning and decision-making

Design data

- Topology **configuration-based** descriptions, generated netlists, and simulated specs
- Can support instruction guided tasks: topology connection, specs range based topology selection

Experimental Takeaways

Model	Topology Generation Solve Rate	Sizing Performance (Low Target)
Gpt-4o-mini	16/20	11/20
O1-mini	16/20	11/20
O3-mini	16/20	11/20

- Improved Generalization:
  - LLM generalizes beyond a few-shot prompt with limited topology connection examples
  - Successfully generates new topologies, not just repeating known ones
- Improved Reliability:
  - Fewer tokens needed → fewer errors
  - More consistent and stable generation
- Performance:
  - Parameter sizing optimization by LLM

Limitations & Future Work

- **Limitations**
  - Parameter size is blind and inefficient
  - Operation point is difficult to locate when more than three active transistors are involved
  - Cannot handle differential-input canonical forms due to strict constraints

- **Further Work**
  - Guided bias circuit generation
  - Integrate optimization algorithms and domain knowledge assistance (expanding reasoning data)
  - Relax constraints and utilize generated database to support a broader range and higher hierarchy of circuit category
  - Collect more diverse data for future training tasks

School of Engineering | MIT Schwarzman College of Computing

MTL MICROSYSTEMS TECHNOLOGY LABORATORIES

LLM Agent Assisted Workflow for Analog Circuit Design Data Collection

Yan Xu[1], Wenjie Lu[2], Tao Yu[2], Ruonan Han[1]

[1]Department of EECS, MIT

[2]Analog Device Inc.

https://hangroup.mit.edu

MIT AI Hardware Program