Accurate and Consistent Graph Model Generation from Text with Large Language Models

Boqi (Percy) Chen[1*], Ou Wei[2], Bingzhou Zhen[2], Gunter Mussbacher[1]

[1]Electrical and Computer Engineering, McGill University, Canada
[2]Huawei Research Canada, Canada
[*]Work partially done during an internship at Huawei Research Canada

Manual Model Generation

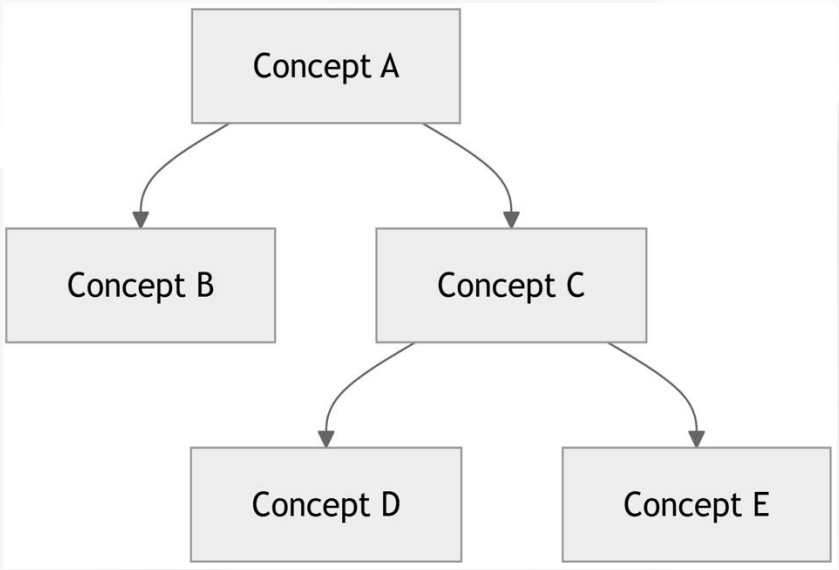
Textual description
e.g., use case description

Graph model
e.g., flowchart

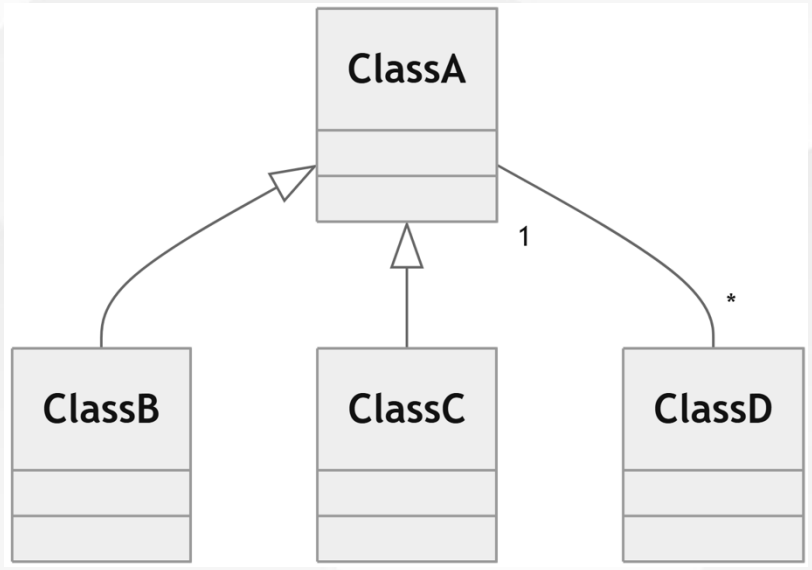
Manual generation can be **time-consuming** and **error-prone**

Graph Models

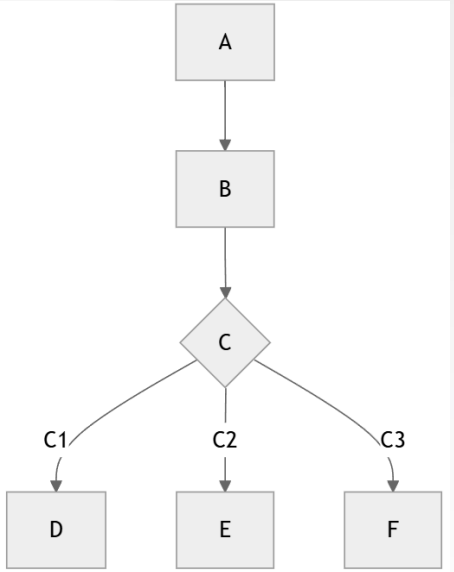
Many types of graph models are used in MDE processes



Taxonomies



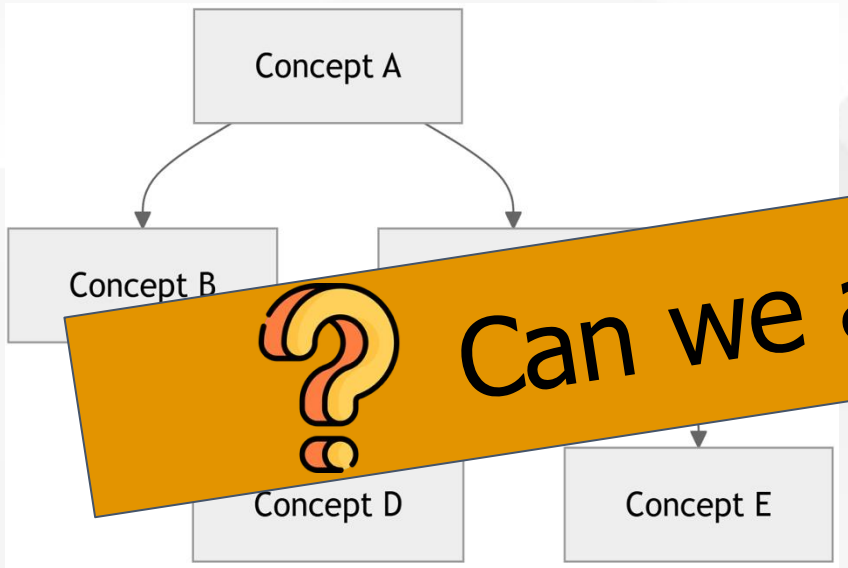
Domain models



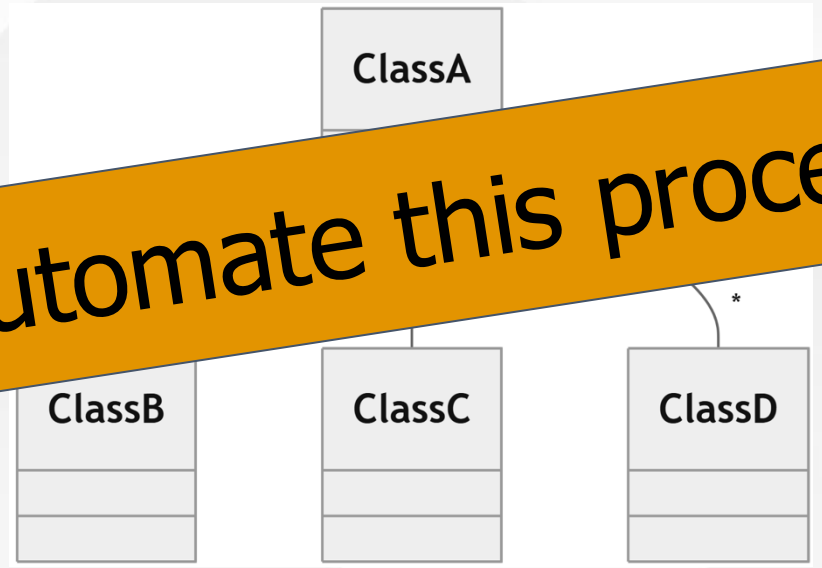
Flowcharts

Graph Models

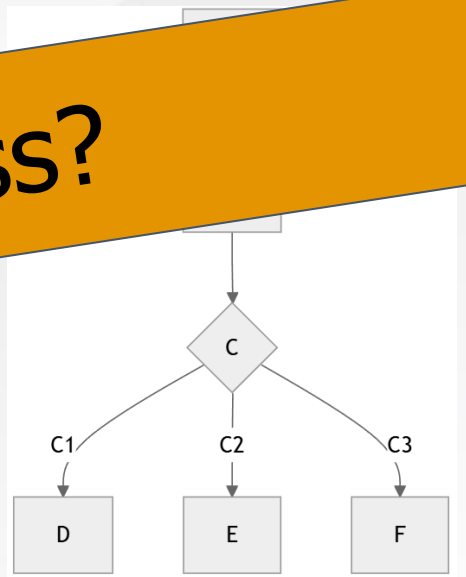
Many graph models are used in many SE processes



Taxonomies



Domain models



Flowcharts

Large Language Models

Maybe we can use LLMs?

Can LLM generate good graph models?

Good Graph Models

No **syntax error**

Consistent

Accurate

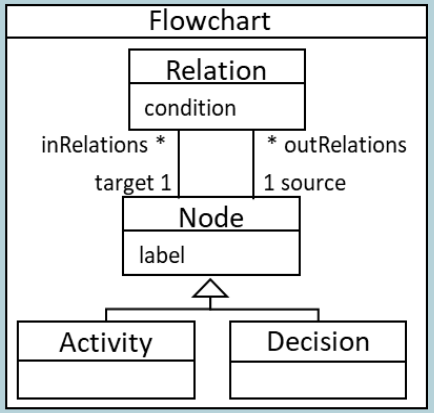
Metamodel

Constraints

Textual description

LLM Generation

Metamodel



Constraints

1. There is only one starting node
2. There is no self-loop
3. Decision nodes must have conditions

Textual description

Description about activity A, B, D and condition C

Can LLM generate good graph models?

Syntax

Filtering

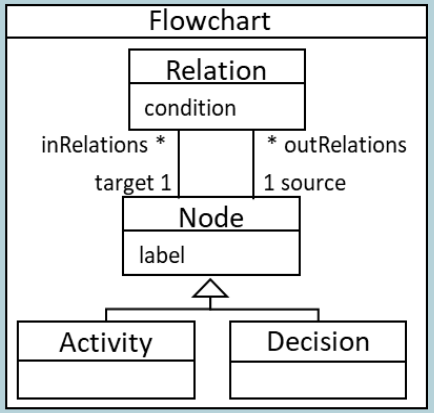
Constrained decoding

Accurat… …sistent?

LLM	F1-score	% Consistent
GPT-4o	79.13	96.63
Llama 3.1-70b	79.54	93.25

Leverage Self Consistency

Metamodel



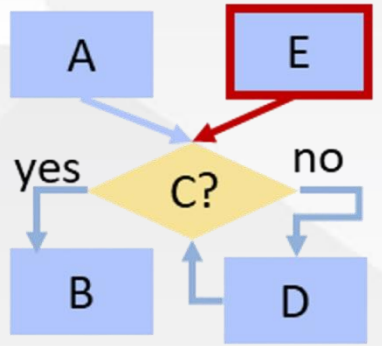
Constraints

1. There is only one starting node
2. There is no self-loop
3. Decision nodes must have conditions

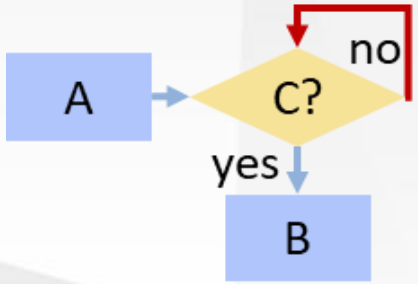
Textual description

Description about activity A, B, D and condition C

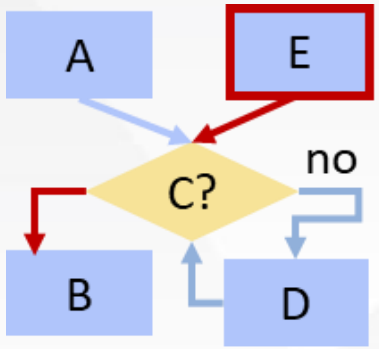
Syntax error



Extra activity E



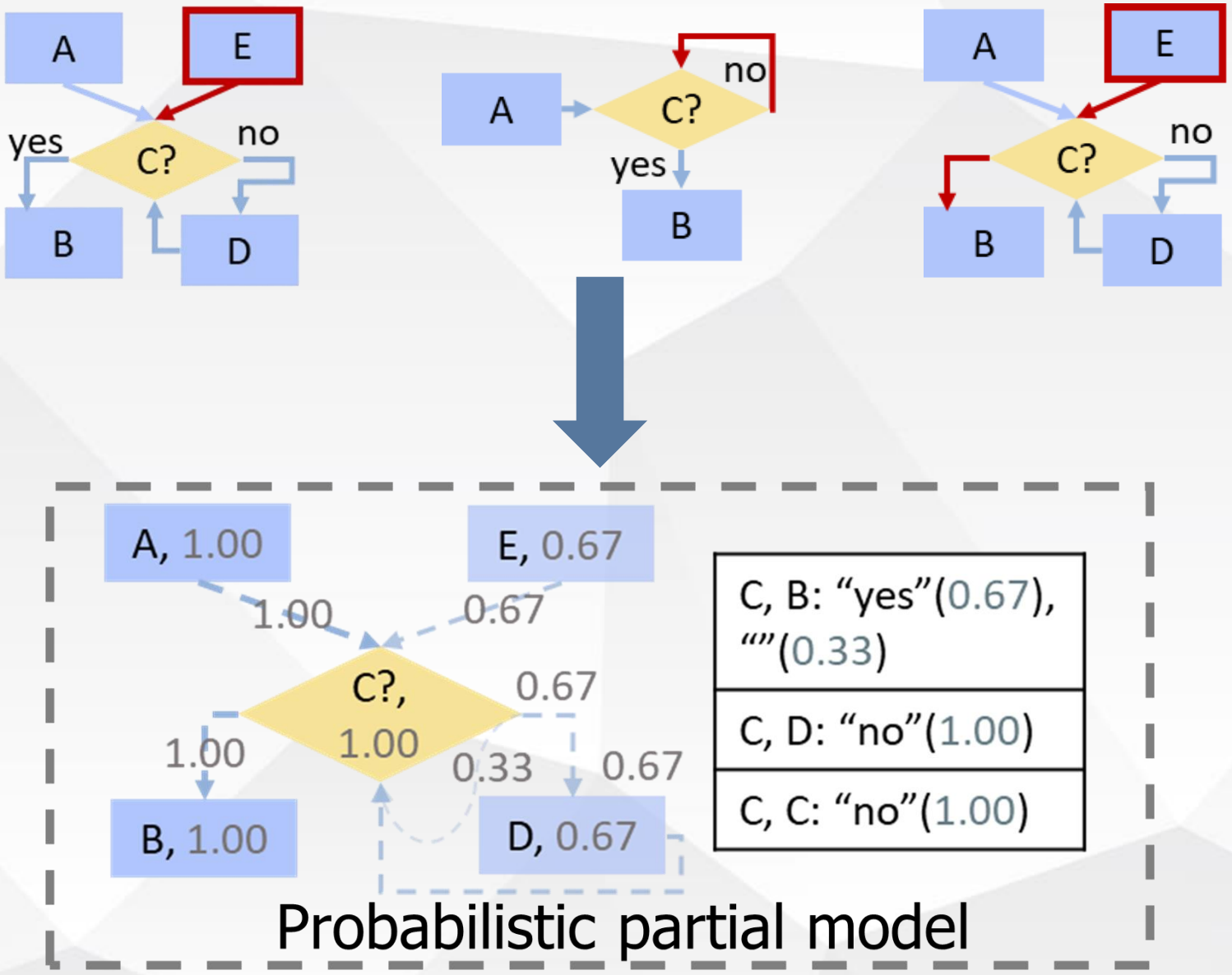
Self loop



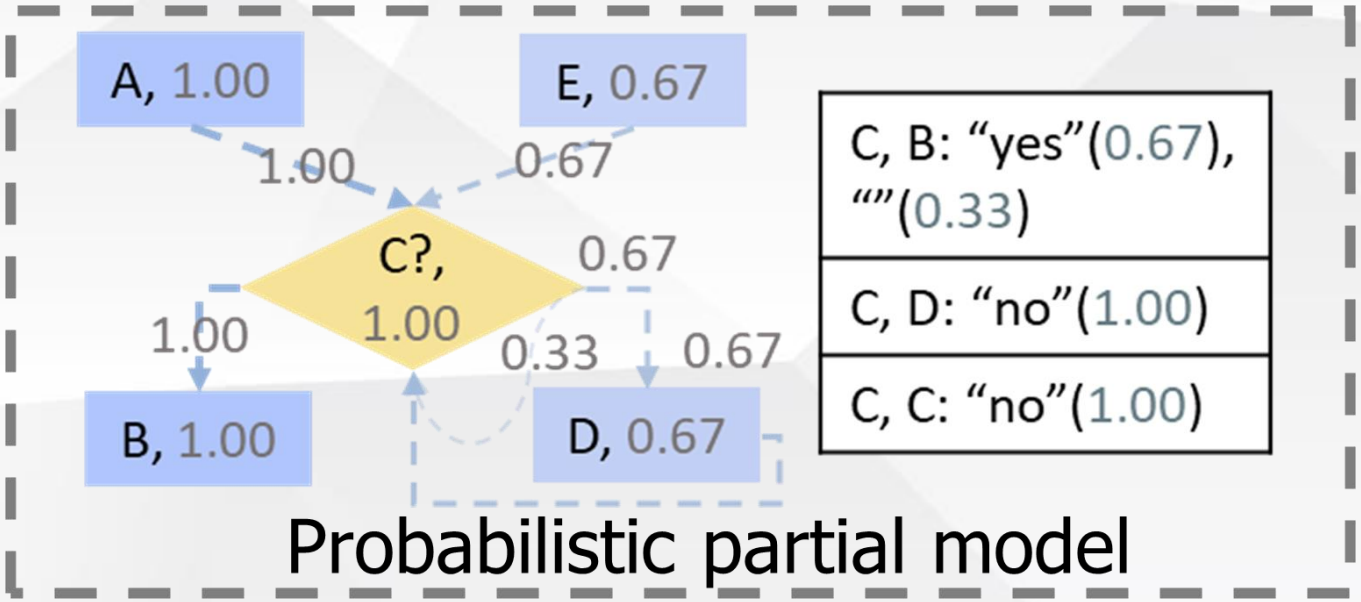
No condition

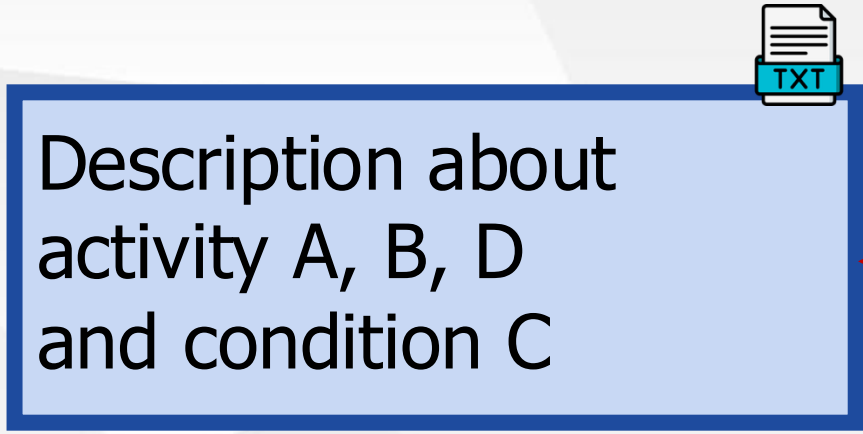
Abstraction

Abstraction with **node similarity** and **graph matching**

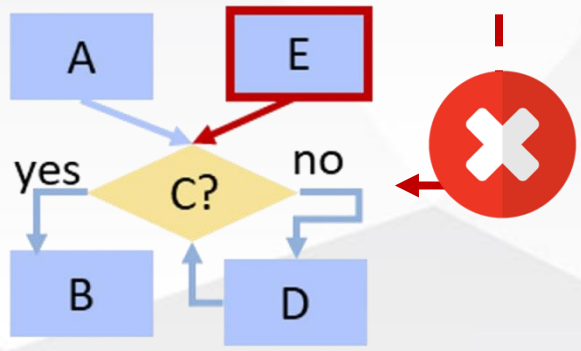


Concretization





Maximize probability only
Maximum likelihood



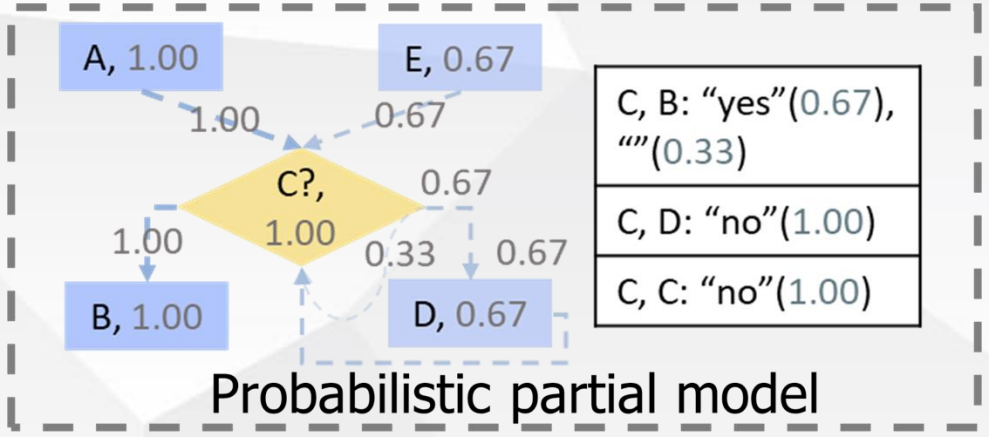
Concretization

Constraint translation

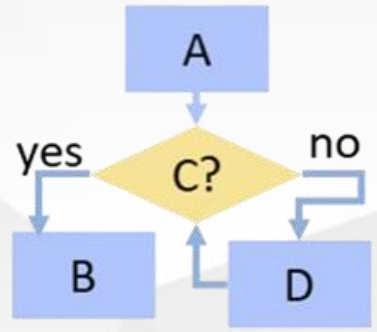
1. ∀Node u,v;!inRelations(u,*)∧ !inRelations(v,*)⇒u=v
2. ∀Node u,v; Next(u,v)⇒u≠v
3. ∀Relation r,Decision d; outRelations(d,r)⇒r.condition≠""

Problem formulation

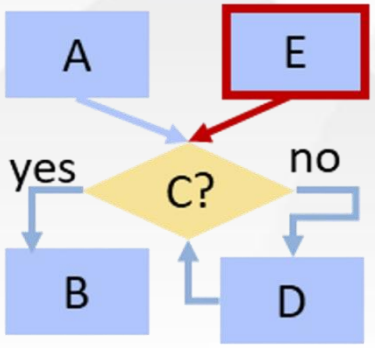
Optimization

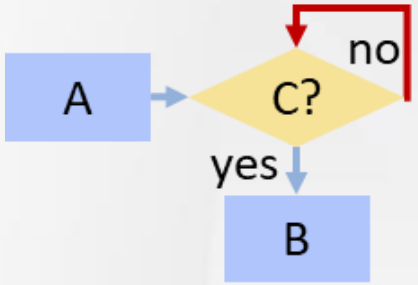


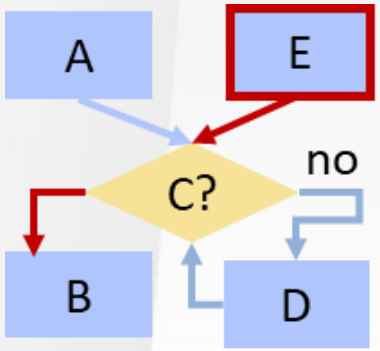
Constraint optimization



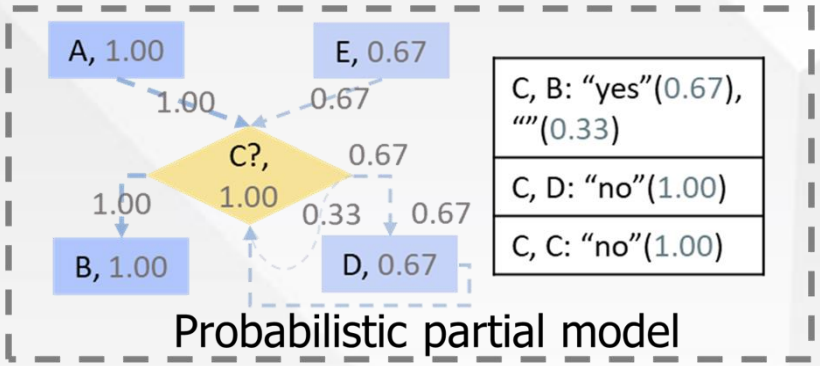
Abstraction Concretization



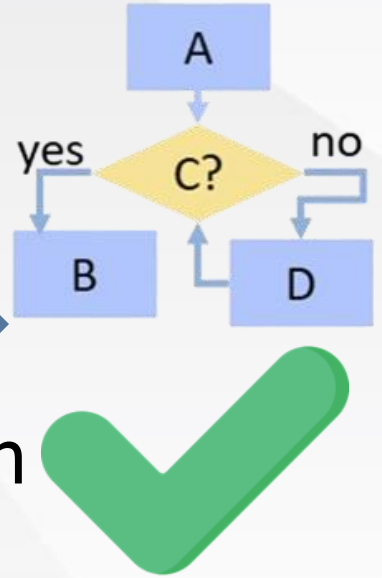




Abstraction



Concretization



Performance Evaluation

Model types


Flowcharts


Taxonomies


Executable models

Metrics

- **Accurate:** *F1* score, execution accuracy (*ACC*)
- **Consistent:** % of consistent models (*Con*), execution success rate (*SR*)

Setup: 10 candidates

AbsCon generates **more accurate and near perfect consistent** models

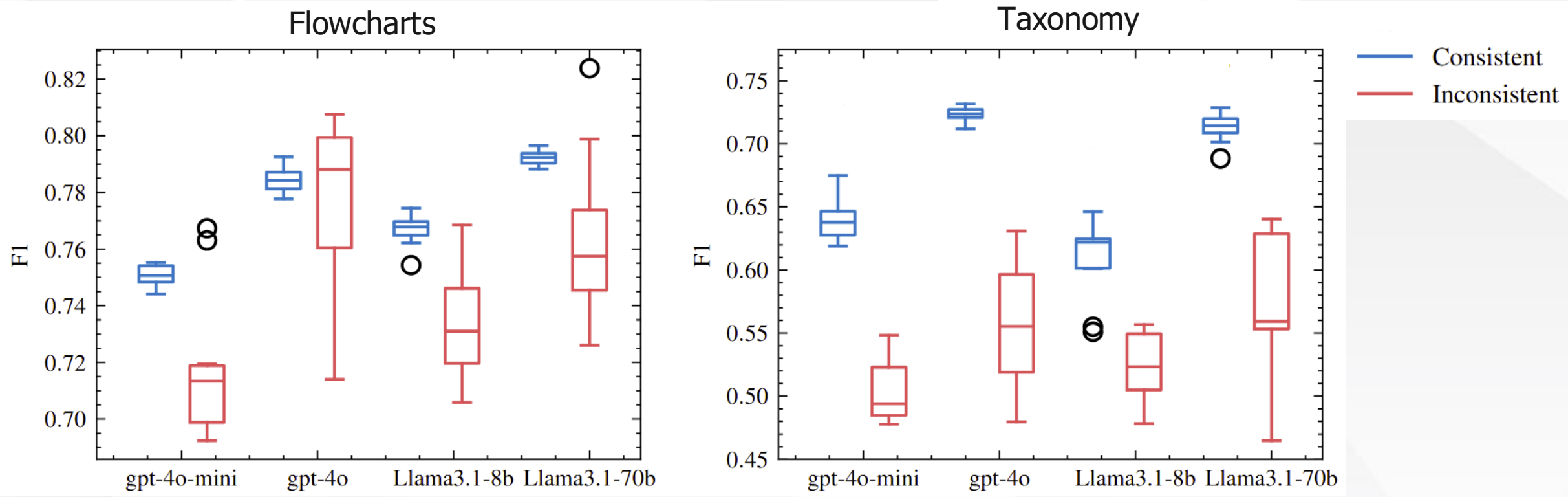

F1 ↑ 0.78%
Con ~ 98.31%


F1 ↑ 8.61%
Con ~ 99.75%

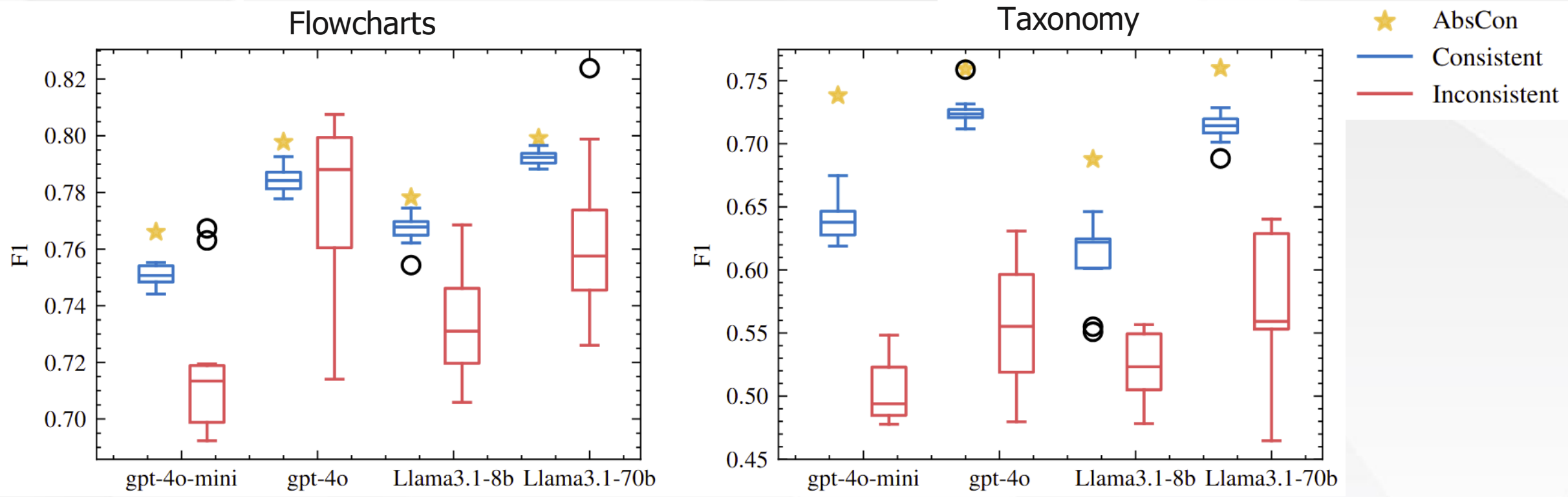

ACC ↑ 27%
SR ~ 99.58%

Smaller LLMs (e.g., GPT-4o-mini) with AbsCon **can beat larger LLMs** (e.g., GPT-4o) baselines

Why more accurate?



Why more accurate?



Future work

AbsCon can still be improve w.r.t. the (theoretically) best candidate

Abstraction

- Candidate generation
- Model matching
- Probability estimation

Concretization

- Constraint derivation
- Objective function
- Efficient graph solver

Thank you! Questions?

Uncertainty → Quality

Paper artifacts

Paper preprint

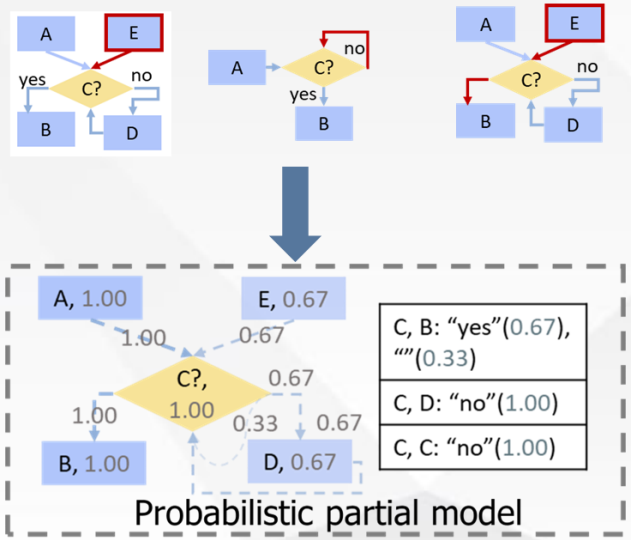
Textual description
e.g., use case description

Graph model
e.g., flowchart

Manual generation can be **time-consuming** and **error-prone**

Abstraction

Abstraction with **node similarity** and **graph matching**



Concretization

Constraint translation

1. ∀Node u,v;!inRelations(u,*)∧ !inRelations(v,*)⇒u=v
2. ∀Node u,v; Next(u,v)⇒u≠v
3. ∀Relation r,Decision d; outRelations(d,r)⇒r.condition≠""

Problem formulation

Optimization

COIN-OR

GUROBI OPTIMIZATION