

LLM-Powered Automatic Theorem Proving and Synthesis for Hybrid Systems and Games

Aditi Kabra

Carnegie Mellon University
(now Mistral AI)

Jonathan Laurent

Karlsruhe Institute of
Technology
(now Axiom Math)

Ruben Martins

Carnegie Mellon University

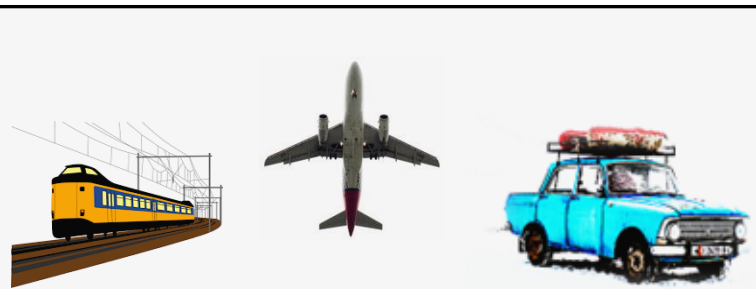
Stefan Mitsch

DePaul University

André Platzer

Karlsruhe Institute of
Technology

Hybrid Game Verification/Synthesis



Hybrid Systems

```

[1] == true; stopOnRaise(r:=t; break;
o=u.length; r&&(s=t,c(r)) return this;
ction(){return u=[],this,disable:fun
e:function(){return p.fireWith(this,a
nding",r={state:function(){return n},a
omise)?e.promise().done(n.resol
d(function(){n=s,t[l^e][2]
0,n=h.call(arguments),r=n.length
r),l=Array(r);r>t;t++n[t]&&b.lunct
>table</table><a href='a'>a</a><im
TagName("input")[0],r.style.cssText="v'
est(r.getAttribute("style")),hrefNorm

```

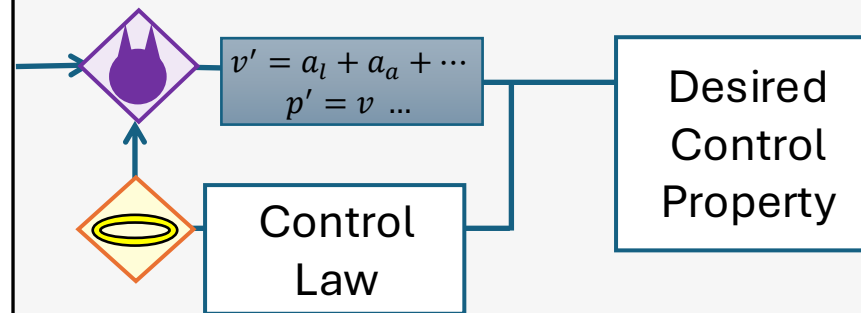
$$v' = a_l + a_a + a_s(p) + a_r(v) + a_c(p)$$

$$p' = v, v' = a_l + a_a + a_s(p) + a_r(v) + a_c(p)$$

$$\left(\sqrt{4(a_l + m_s)a_2 - a_1^2} \right) \cdot$$

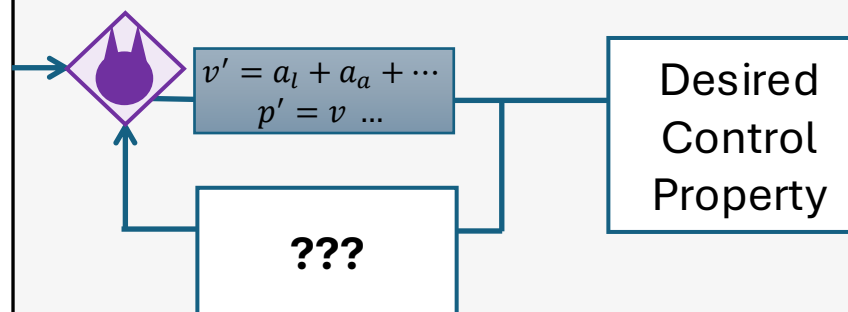
$$p = \frac{\tan \left(t \frac{\sqrt{4(a_l + m_s)a_2 - a_1^2}}{2} + \tan^{-1} \left(\frac{a_1 + 2a_2 v_0}{\sqrt{4(a_l + m_s)a_2 - a_1^2}} \right) \right) - a_1}{2a_2}$$

Verify Control Conditions



Automatically check if the control system is correct given possible **continuous, discrete,** and **adversarial** dynamics

Synthesize Correct Control Conditions



Automatically generate control principles with **mathematical guarantees** of correctness

Automated Reasoning Tools



Deductive logics:

- ✓ precise reasoning free of discretization or approximation
- ✓ Over nonlinear dynamics interacting with discrete control
- ✓ in terms of symbolic parameters
- ✓ cleanly handle infinite state space
- ✓ and unbounded domains including unbounded time horizons

But

- generally require effort-intensive interactive theorem proving

[1] L. Bu, Y. Li, L. Wang, and X. Li, “Bach: Bounded reachability checker for linear hybrid automata,” in Proceedings of the 2008 International Conference on Formal Methods in Computer-Aided Design, ser. FMCAD ’08. IEEE Press, 2008.

[2] A. Kundu, S. Das, and R. Ray, “Sat-reach: A bounded model checker for affine hybrid systems,” ACM Trans. Embed. Comput. Syst., vol. 22, no. 2, Jan. 2023.

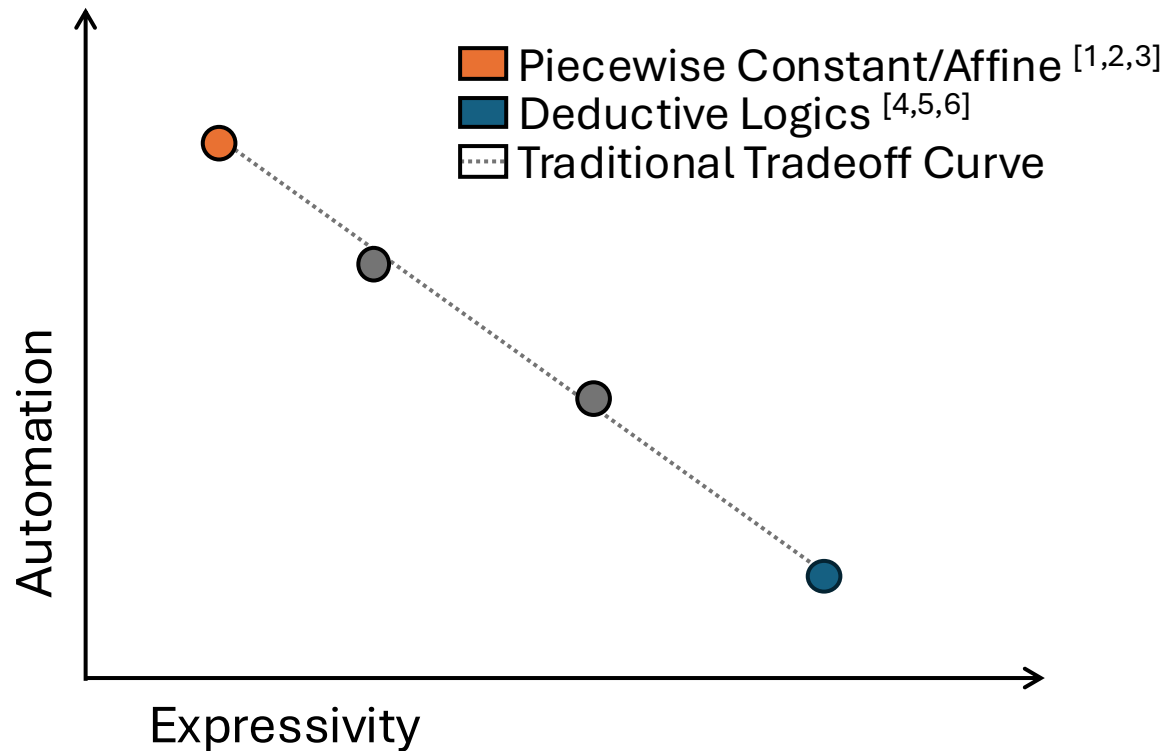
[3] R. Ray, A. Gurung, B. Das, E. Bartocci, S. Bogomolov, and R. Grosu, “Xspeed: Accelerating reachability analysis on multi-core processors,” in Hardware and Software: Verification and Testing - 11th International Haifa Verification Conference, HVC 2015, Haifa, Israel, November 17-19, 2015, Proceedings, ser. Lecture Notes in Computer Science, N. Piterman, Ed., vol. 9434. Springer, 2015, pp. 3–18.

[4] A. Platzer, “Logics of dynamical systems,” in LICS. IEEE, 2012, pp. 13–24.

[5] C. Zhou, J. Wang, and A. P. Ravn, “A formal description of hybrid systems,” in Hybrid Systems III: Verification and Control, Proceedings of the DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, October 22-25, 1995, Rutgers University, New Brunswick, NJ, USA, ser. Lecture Notes in Computer Science, R. Alur, T. A. Henzinger, and E. D. Sontag, Eds., vol. 1066. Springer, 1995, pp. 511–530

[6] S. Wang, N. Zhan, and L. Zou, “An improved HHL prover: An interactive theorem prover for hybrid systems,” in Formal Methods and Software Engineering - 17th International Conference on Formal Engineering Methods, ICFEM 2015, Paris, France, November 3-5, 2015, Proceedings, ser. Lecture Notes in Computer Science, M. J. Butler, S. Conchon, and F. Zaidi, Eds., vol. 9407. Springer, 2015, pp. 382–399.

Automated Reasoning Tools



Deductive logics:

— generally require effort-intensive interactive theorem proving



Complete FRA Model^[7]

>188k proof steps^[8]

2545 lines of proof tactic^[8]

months of PhD student time^[?]

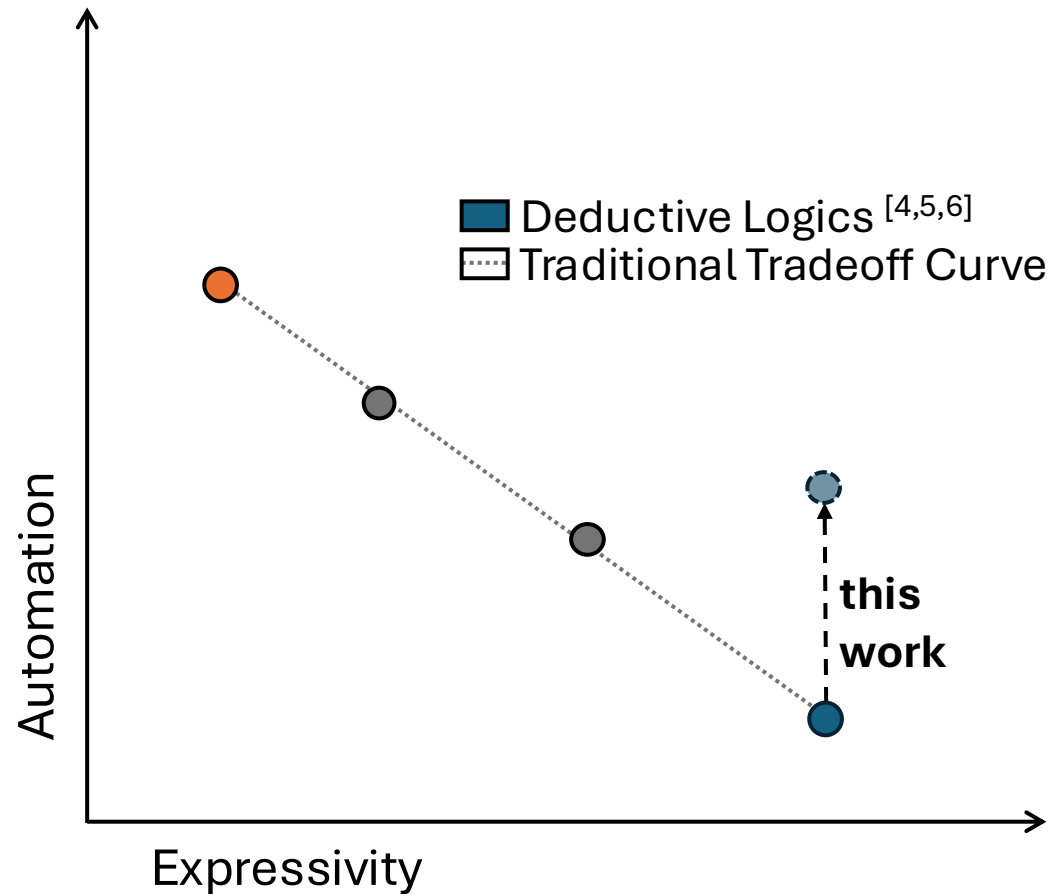
Proof: ✓ All goals closed

```
Provable( ==> end()-trainPos>(vel-min<< ._1 < ._2&
reChangeRate()*vel^(1/2))/pressureChangeRate())*(b0
&._0=-._1|._1>=0&._0=._1 >>(((b0()-maxSlope()-0)^2-2*
essureChangeRate(),(b0()-maxSlope()-0-abs<< ._1 < 0&
0=._1|._1>=._2&._0=._2 >>((Apb()-0)/pressureChangeRa
6*pressureChangeRate()*min<< ._1 < ._2&._0=._1|._1>=
()*vel^(1/2))/pressureChangeRate())^3&(a0()>0&b0())>
ainPos (Rmin())<=curvature(trainPos)&curvature(trainPo
((slopeAcc(x))'<=maxVertCur()*x'&-(slopeAcc(x))'<=max
Pos>(vel+(a0()+min<< ._1 < ._2&._0=._1|._1>=._2&._0=
Der()*(vel+(a0()+maxSlope()*T())*T(),0))*T()-min<<
2-2*pressureChangeRate()*(vel+(a0()+min<< ._1 < ._2&
trainPos)+crvDer()*(vel+(a0()+maxSlope()*T())
geRate(),(b0()-maxSlope()-0-abs<< ._1 < 0&._0
(slope()*T()),maxSlope()))+
+(vel+(a0()+min<< ._1 < ._2&._0=._1|._1>=._2&._0=
```

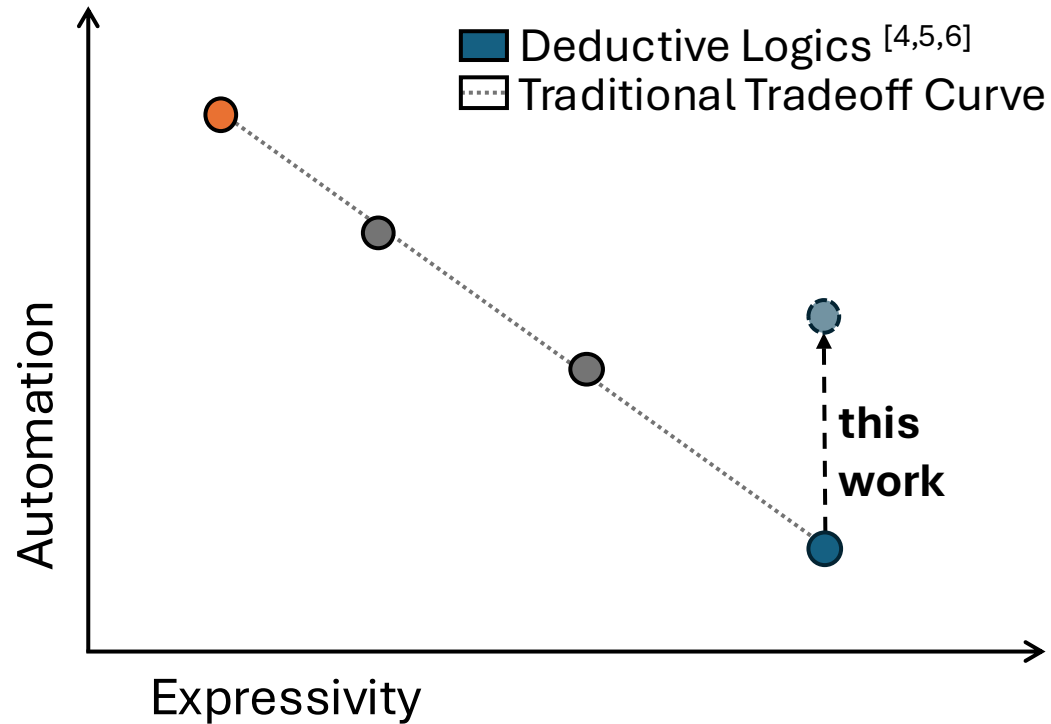
[7] J. Brosseau and B. M. Ede, “Development of an adaptive predictive braking enforcement algorithm”, Federal Railroad Administration, 2009.

[8] Verified Train Controllers for the Federal Railroad Administration Model: Balancing Competing Brake and Track Forces. Aditi Kabra, Stefan Mitsch, André Platzer. EMSOFT 2022.

But with LLMs?



But with LLMs?



This has been done for:

- Math [9,10,11]
- Software Verification [12]

This paper: Hybrid Systems/Games

- Data-scarce (maybe ~10 systems)
- LLMs Used *on top of* traditional automation
- Methodology: five hard problems
 - 0/5 without LLMS
 - 5/5 for verification with LLMs
 - 4/5 for synthesis with LLMs

[9] S. Polu and I. Sutskever, “Generative language modeling for automated theorem proving,” ArXiv, vol. abs/2009.03393, 2020.

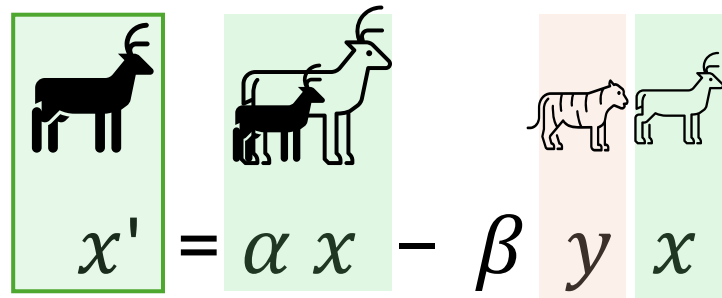
[10] S. Varambally, T. Voice, Y. Sun, Z. Chen, R. Yu, and K. Ye, “Hilbert: Recursively building formal proofs with informal reasoning,” in NeurIPS Workshop, 2025.

[11] K. Yang, A. M. Swope, A. Gu, R. Chalamala, P. Song, S. Yu, S. Godil, R. J. Prenger, and A. Anandkumar, “Leandojo: Theorem proving with retrieval-augmented language models,” NeurIPS 2023.

[12] J. Qin, A. Du, D. Zhang, M. Lentz, and D. Zhuo, “Can large language models verify system software? a case study using fscq as a benchmark,” HotOS ’25.

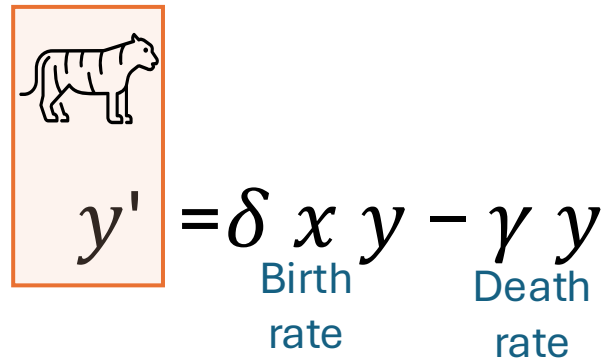
Verification Example: Lotka–Volterra

A forest department must keep prey and predator populations above safe minima.


$$x' = \alpha x - \beta y x$$

Birth
rate

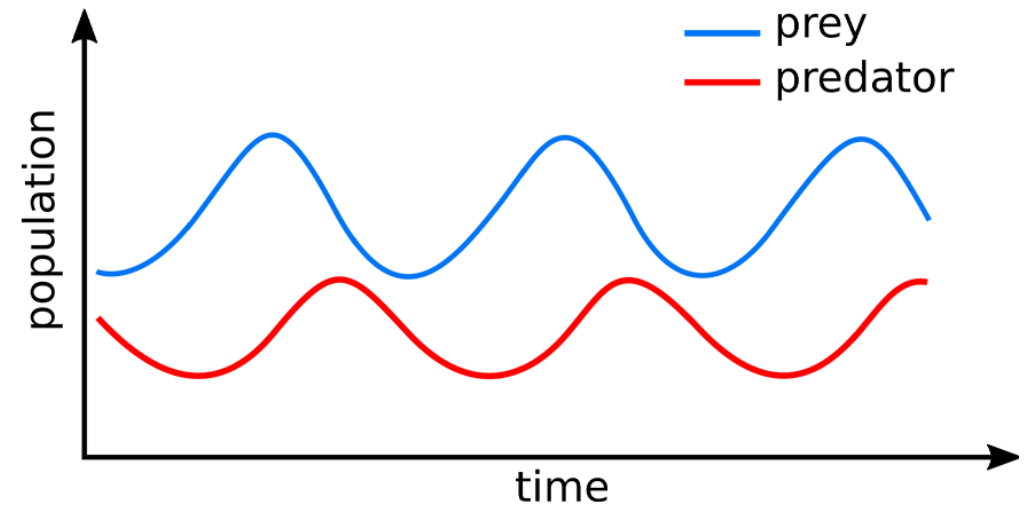
Death
rate


$$y' = \delta x y - \gamma y$$

Birth
rate

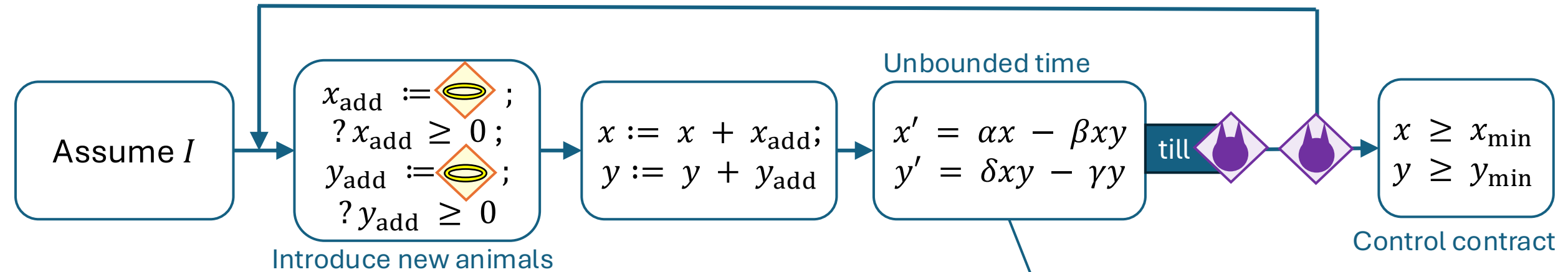
Death
rate

Nonlinear dynamics



Verification Example: Lotka–Volterra

A forest department must keep prey and predator populations above safe minima.



$I: \alpha, \beta, \delta, \gamma > 0; x, y > 0; x_{min} \leq x \leq \gamma/\delta; y_{min} \leq y \leq \alpha/\beta$

($\exists R$) $\frac{\Gamma \vdash p(e), \Delta}{\Gamma \vdash \exists x p(x), \Delta}$ (arbitrary term e)

(loop) $\frac{\Gamma \vdash J, \Delta \quad J \vdash [\alpha]J \quad J \vdash P}{\Gamma \vdash [\alpha^*]P, \Delta}$

LLM identifies 24-line tactic script

(dRI) $\frac{\Gamma, Q \vdash \bigwedge_i^N \bigwedge_j \mathcal{L}_p^i(h_j) = 0}{\Gamma \vdash [x' = p \ \& \ Q] \bigwedge_j h_j = 0}$

*

$$\frac{\frac{x - g/d = 0 \quad \bigwedge y - a/b = 0 \vdash d(ax - bxy) = 0 \quad \bigwedge b(dxy - gy) = 0}{x - g/d = 0 \quad \bigwedge y - a/b = 0 \vdash [x' = ax - bxy, y' = dxy - gy]}}{dRI} \quad (x - g/d = 0 \wedge y - a/b = 0)$$

```

unfold;
loop("x <= g/d & y <= a/b & g/d
    >= xmin &
        a/b >= ymin & x > 0 &
        y > 0 & a > 0 &
        b > 0 & d > 0 &
        g > 0 & x >= xmin &
        y >= ymin", 1);
<(
    QE("Z3"),
    unfold;
    existsR("g/d-x", 1);
    unfold;
    <(

```

```

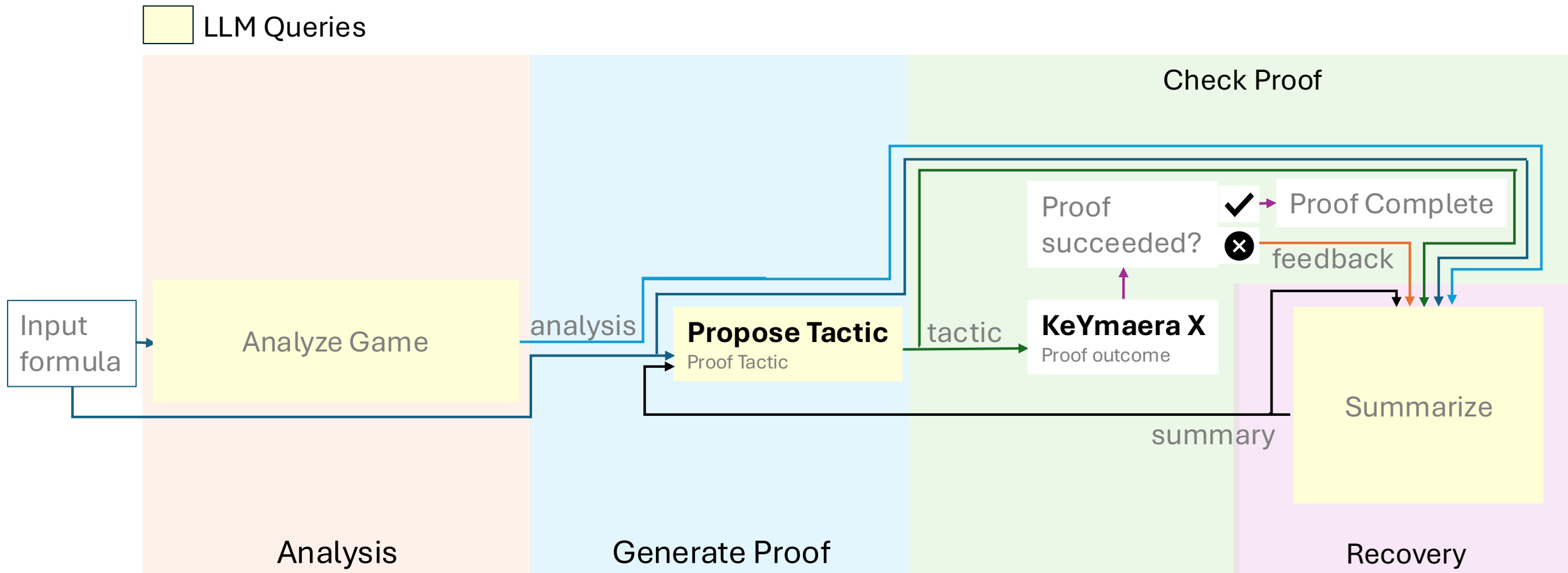
    QE("Z3"),
    existsR("a/b-y", 1);
    unfold;
    <(
        QE("Z3"),
        unfold;
        print("State before ODE cut after
                choosing xadd and yadd");
        dC("((d*x-g)^2+(b*y-a)^2)=0", 1);
        <(
            dW(1); QE("Z3"),
            dbx("2*(d*x-g)*(b*y-a)*(b*y-d*x) /
                ((d*x-g)^2+(b*y-a)^2)", 1)
        )
    )
    QE("Z3")
)

```

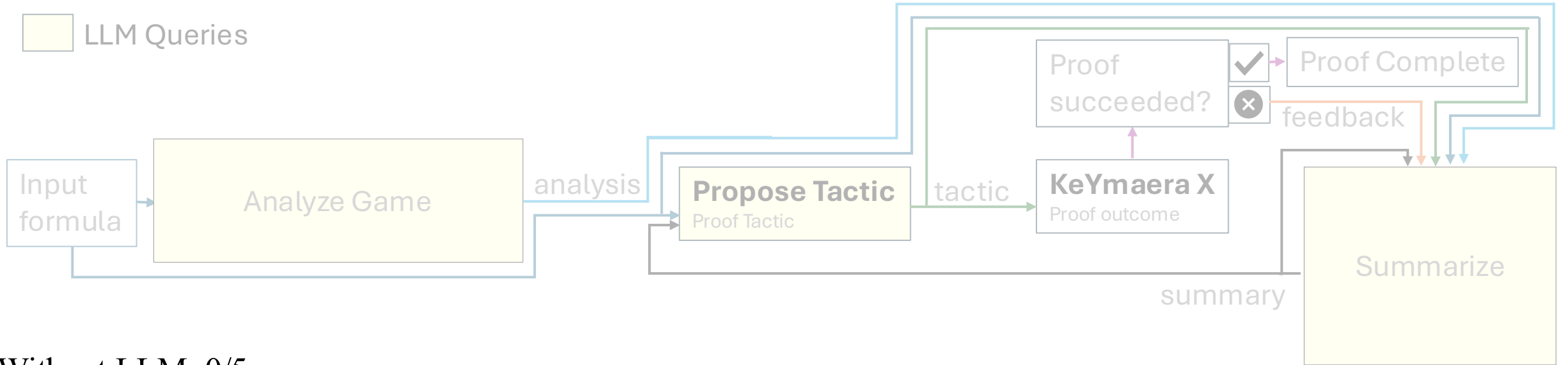
Five Case Studies

Case study	Control objective	Reasoning challenge
Lotka–Volterra	Maintain minimum prey and predator populations	Unbounded time reasoning over nonlinear dynamics
Train	Maintain train safety	Nonlinear dynamics; nested loops and coupled invariants
Chemical Reaction	Keep reaction temperature within safe limits	arbitrarily alternating control modes over unbounded iterations, dynamics outside the decidable (polynomial) fragment
Coolant	Meet heat-absorption quotas while limiting coolant discharge	Complex, timed reach-avoid objective; qualitative behavior changes by initial state
Van der Pol	Choose initial conditions that keep the oscillator safe	Nonlinear oscillations over an unbounded time horizon

Verification Pipeline



Verification Pipeline



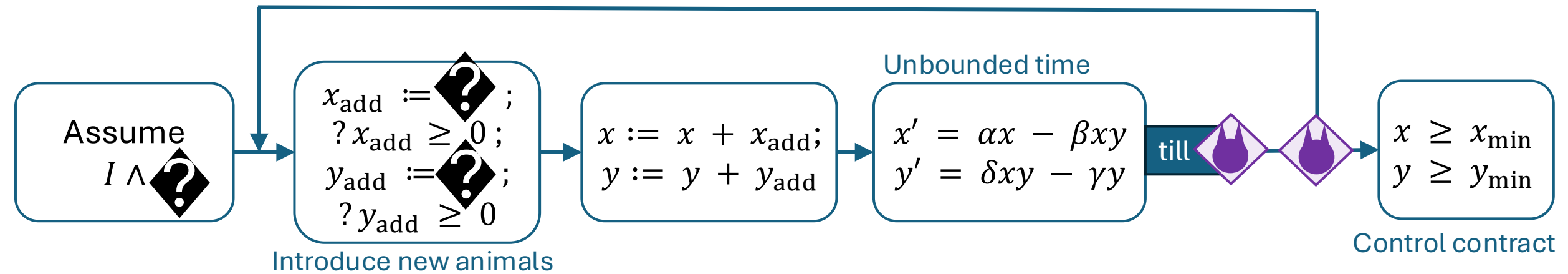
Without LLM: 0/5

With LLM: 5/5

Model	Reasoning	Solved	Pass@1	Avg Cost (\$)	Avg Calls	Avg Output (k Tokens)
Qwen3.5	✓	1/5	0.13 ± 0.13	—	88.0 ± 18.0	254 ± 17
GPT-5.4-mini		1/5	0.13 ± 0.13	1.07 ± 1.05	138.5 ± 130.5	41 ± 39
GPT-5.4-mini	✓	2/5	0.33 ± 0.21	0.70 ± 0.17	11.0 ± 2.7	148 ± 36
GPT-5.4	✓	5/5	0.93 ± 0.07	2.21 ± 0.44	9.4 ± 1.4	92 ± 18
GPT-5.5		3/5	0.60 ± 0.24	5.95 ± 5.25	47.3 ± 33.7	35 ± 30
GPT-5.5	✓	5/5	1.00 ± 0.00	2.05 ± 0.56	6.1 ± 1.3	63 ± 17

Synthesis Example: Lotka–Volterra

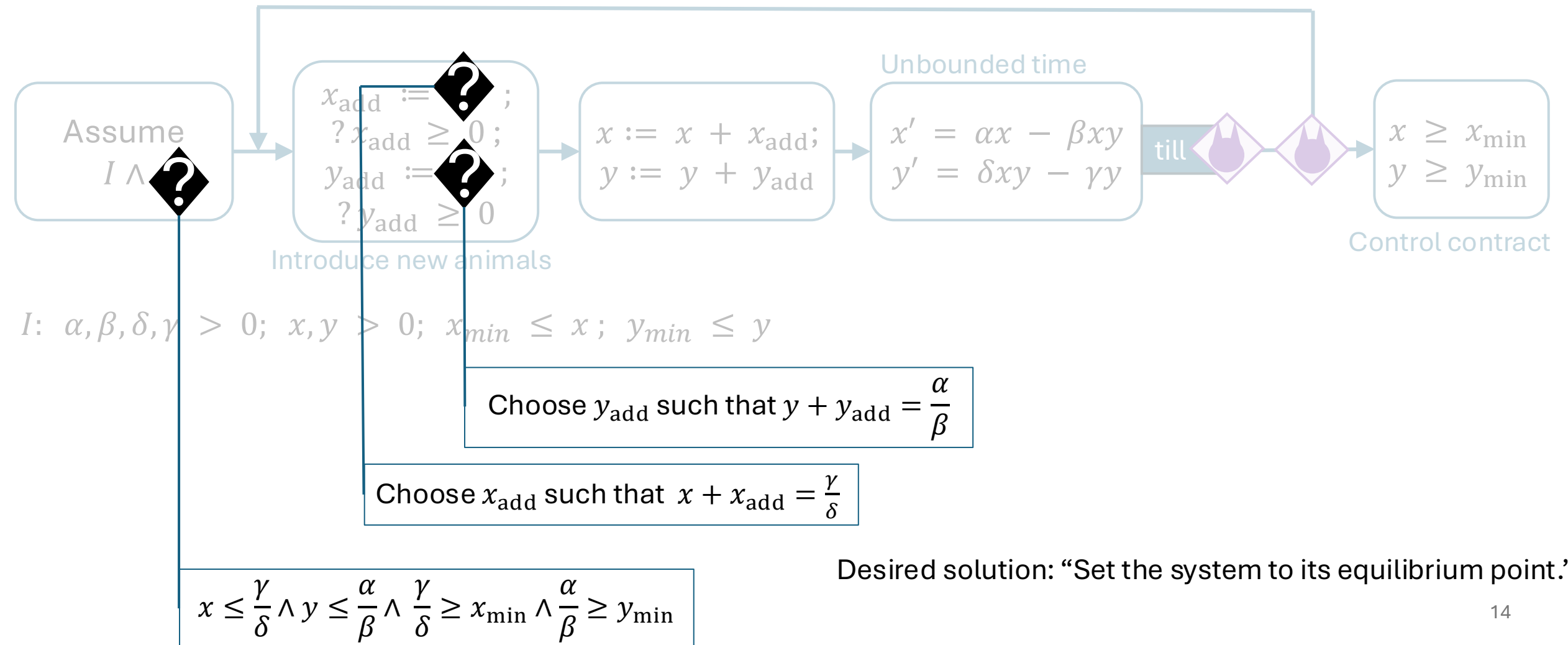
A forest department must keep prey and predator populations above safe minima.



$I: \alpha, \beta, \delta, \gamma > 0; x, y > 0; x_{min} \leq x; y_{min} \leq y$

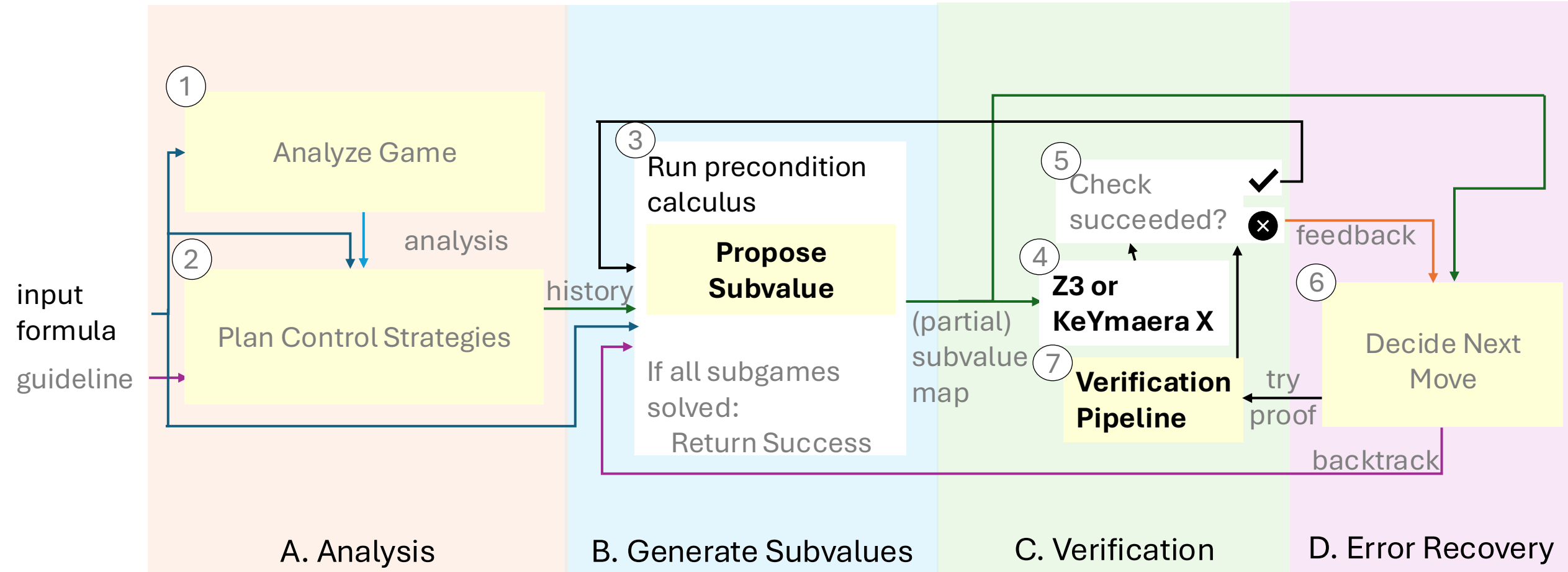
Synthesis Example: Lotka–Volterra

A forest department must keep prey and predator populations above safe minima.

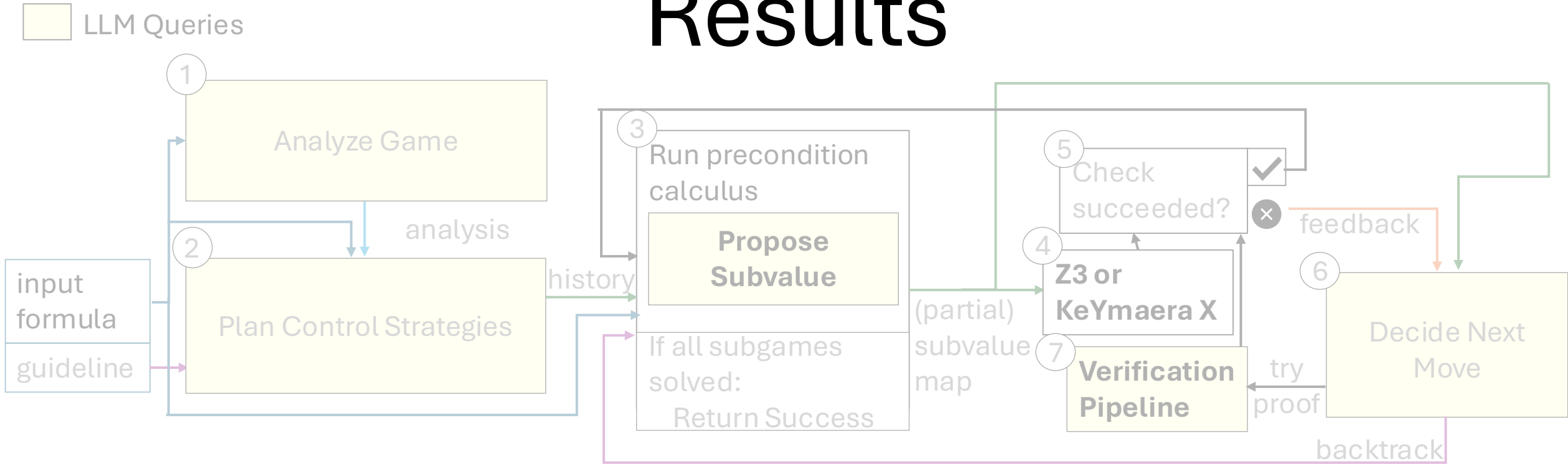


Synthesis Pipeline

LLM Queries



Results



Results (GPT 5)

Configuration	Solved	Pass@1	Avg Cost (\$)	Avg Calls	Avg Output (k Tokens)
Baseline	0/5	—	—	—	—
Ablation(-⑥,⑦)	2/5	0.40 ± 0.24	0.80 ± 0.06	19.5 ± 2.5	76 ± 7
Full Pipeline	4/5	0.80 ± 0.20	2.19 ± 0.73	41.0 ± 10.0	203 ± 67

Conclusion

- New class of class of problems reached by automatic proving
AI + deductive proving
- Could unblock broader use cases/adoption