Solving the Curse of Dimensionality in DDCM

Graph-Based Computation for Activity-Based Travel Demand Models

January 2026

Press Space for next page →

1. Why This Matters

The Problem We're Solving

The Policy Questions That Motivate Us

Urban mobility: trams, cars, buses, pedestrians competing for space

🚌 Transit Planning

"If we add a new bus line, who will use it?"

🏠 Remote Work

"How will work-from-home change commute patterns?"

💰 Congestion Pricing

"Who benefits and who loses from road pricing?"

Challenge: Model how people make daily activity and travel decisions.

What Are Travel Demand Models?

ANSWERING POLICY QUESTIONS

To answer questions like "Who will use a new bus line?" or "How will remote work change commutes?", we need models that **predict how people make travel decisions.**

What They Do

- 🎯 Predict **where** people travel
- 🕐 Predict **when** they travel
- 🚗 Predict **how** they travel (mode choice)
- 📊 Forecast impacts of policy changes

The Key Insight

Travel is a **derived demand**—people don't travel for fun, they travel to participate in *activities* (work, shopping, leisure).

→ Modeling **activity choice** gives us the complete picture.

What is DDCM?

Dynamic Discrete Choice Model (Västberg et al., 2020 — *Transportation Science*)

The Core Idea

Your daily schedule is a **sequence of decisions**:

- When should I leave home?
- Should I go to work or run errands first?
- How long should I stay at the office?
- What's the best way to get there?

Key Properties

- ✓ **Economic foundation** — enables welfare analysis
- ✓ **Time consistency** — respects physical constraints
- ✓ **Interdependence** — choices affect future options

Case Study: Workdays in Stockholm, Sweden — 1,240 zones × 4 modes × 6 activities

DDCM as a Markov Decision Process (MDP)

An **MDP** (S, C, q, u) models sequential decision-making under uncertainty.

State (S) — "Where am I now?"

Your current situation:

Time, location, activity, mode, history

Action (C) — "What can I do?"

Available choices:

Stay, travel to destination, change activity

Transition (q) — "What happens next?"

How actions move you to new states:

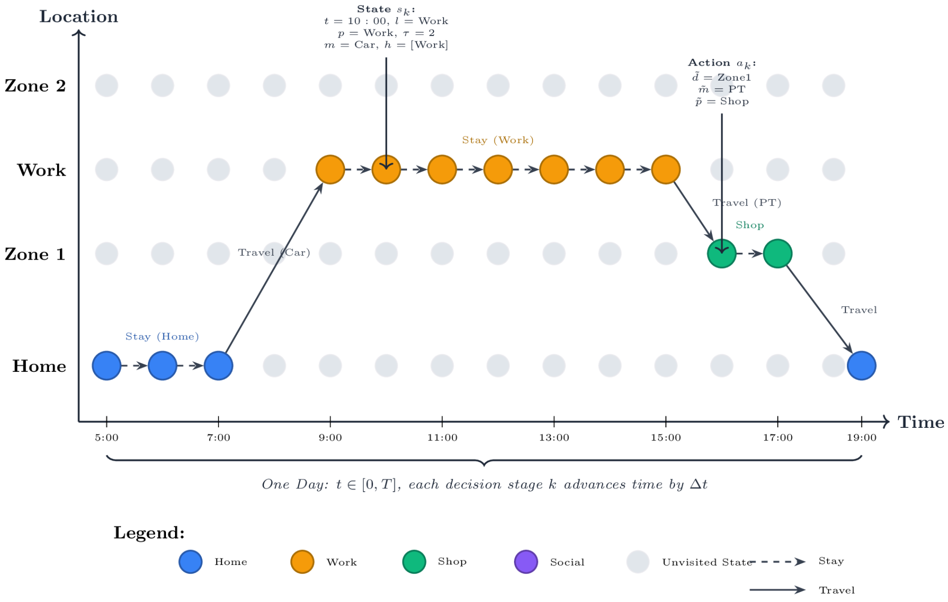
Current state + action → next state

Utility (u) — "How good is this?"

The "reward" for each choice:

Activity enjoyment – travel discomfort

Visualizing a Daily Activity Pattern

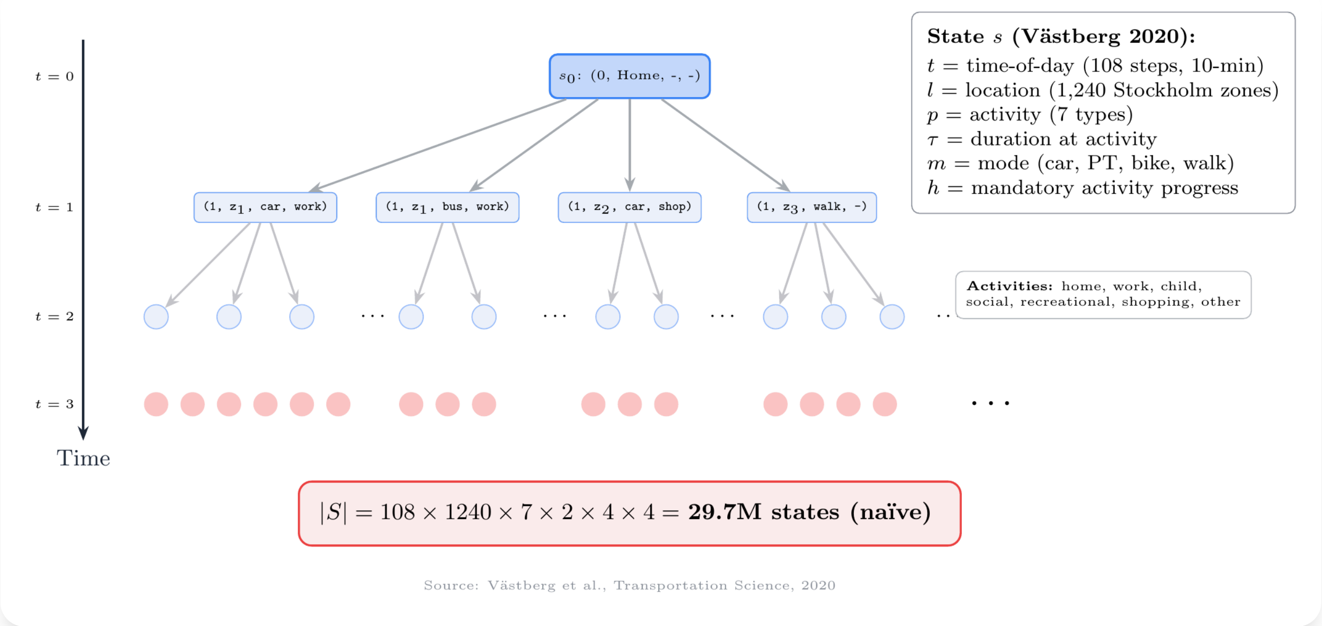


A typical workday: Morning at Home → Travel to Work → Work → Travel to Shopping → Return Home

2. The Problem

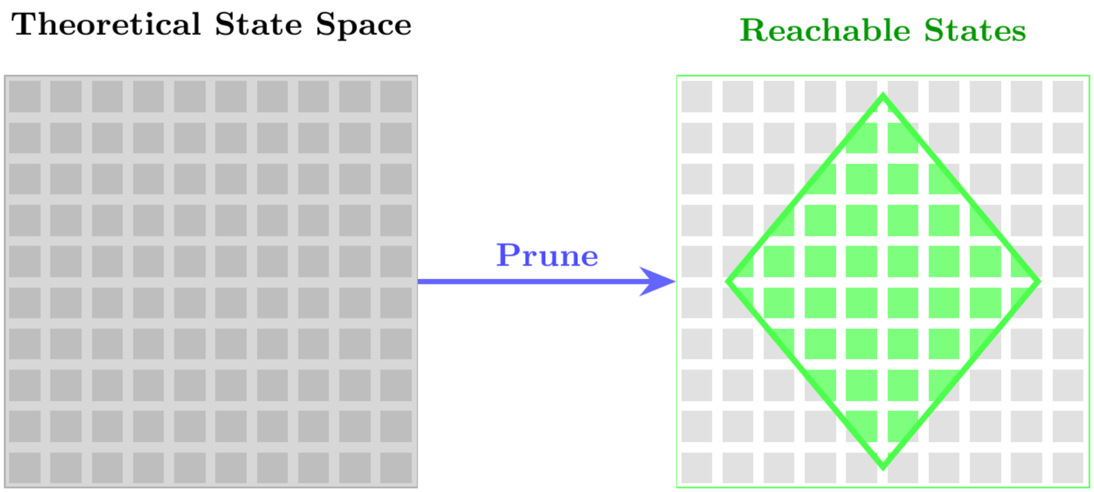
The Curse of Dimensionality

The Curse of Dimensionality



Exponential growth of state space over time

The Key Insight



"What if we don't compute the impossible ones?"

Research Objectives

PRIMARY OBJECTIVE

Make DDCM **computationally tractable** for population-scale simulation.

TARGET OUTCOMES

- ✓ Reduce computational burden by orders of magnitude
- ✓ Enable simulation on consumer hardware (GPU)
- ✓ Preserve theoretical properties

THE CHALLENGE

Current: ~10 seconds per agent → 100,000 agents = **1,000+ CPU-days**

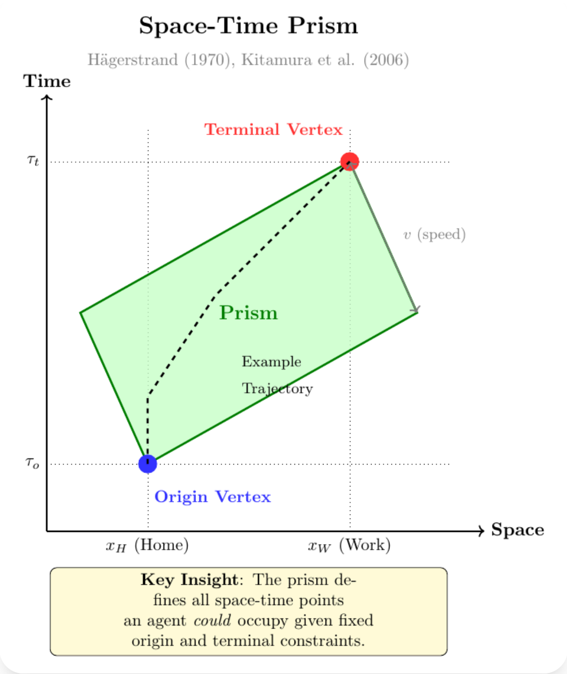
Goal: Make this tractable while maintaining **optimality**

3. Theoretical Foundation

The Reachability Concept

The Space-Time Prism

Hägerstrand (1970): Not everywhere is reachable from everywhere

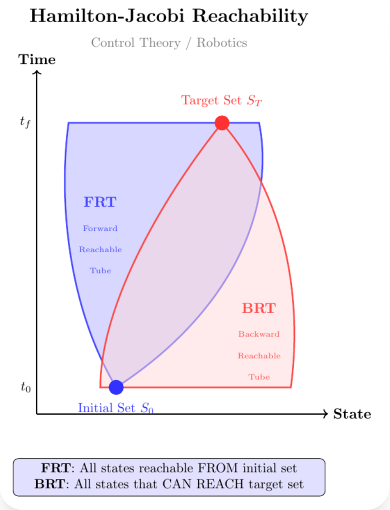


KEY CONCEPT

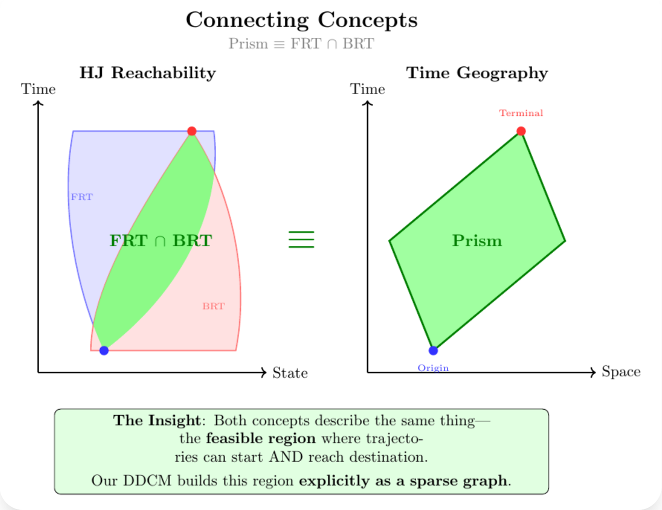
- **Origin**: When/where you can leave
- **Destination**: When/where you must arrive
- **Speed Constraint**: Limits how far you can travel
- **The Prism**: All space-time points you *could* occupy

Activity scheduling has inherent constraints!

Reachability Analysis



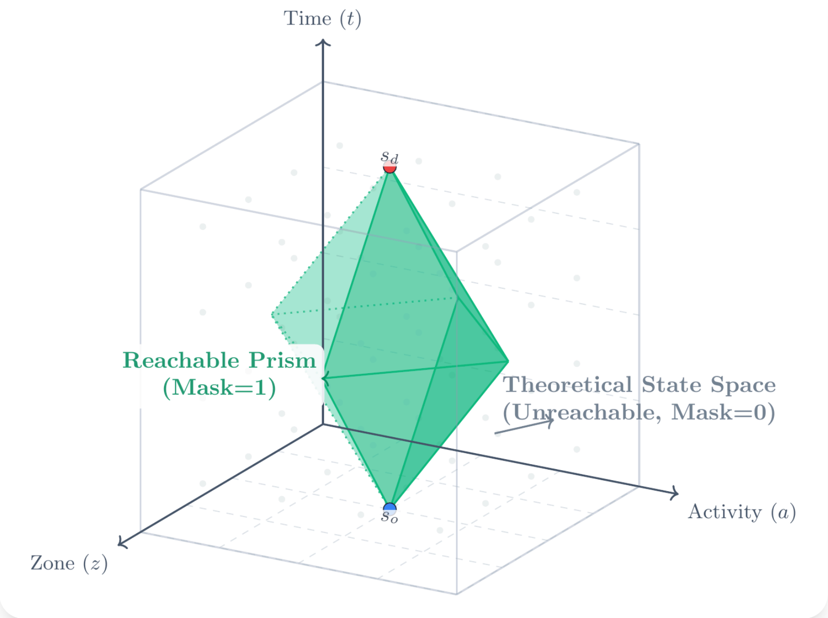
Forward and Backward Reachable Tubes



Intersection = Feasible Region

Prism ≡ FRT ∩ BRT — Only compute where you CAN go AND CAN return from!

Only Compute the Reachable

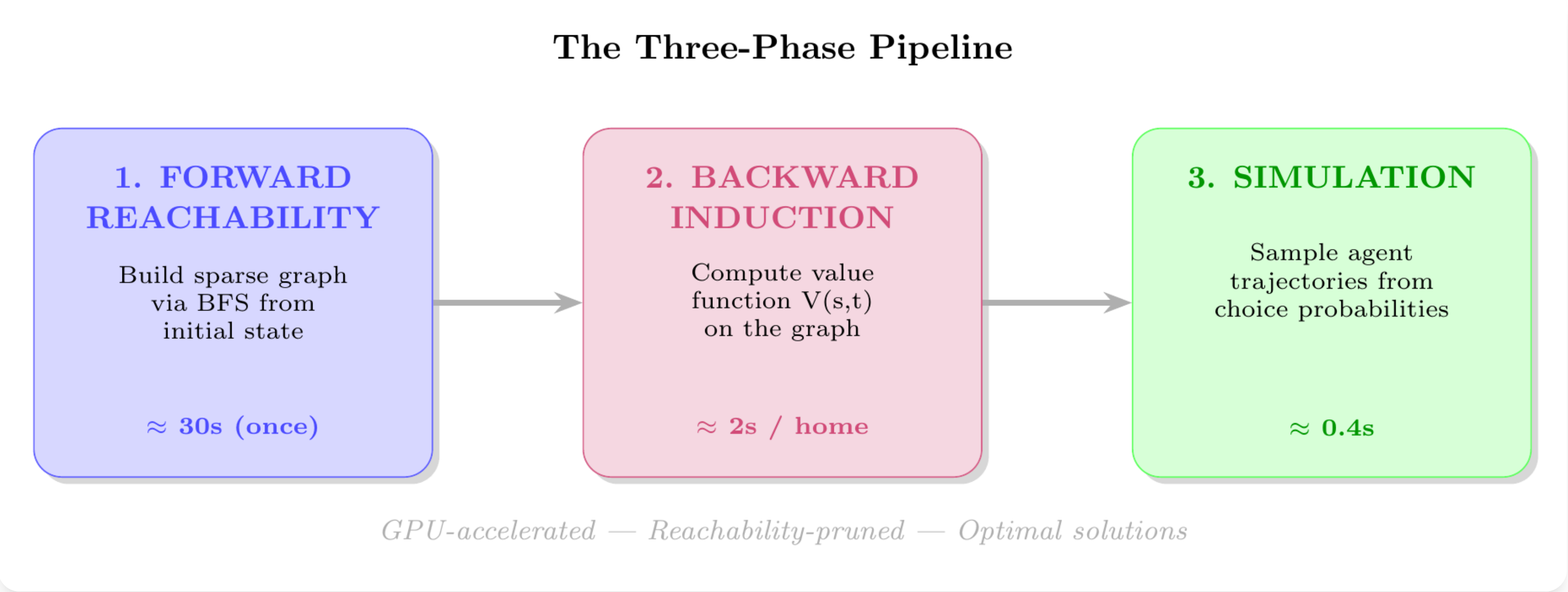


Green Prism (Reachable States) inside **Gray Cube** (Theoretical State Space)

4. Our Solution

The Three-Phase Pipeline

The Three-Phase Pipeline



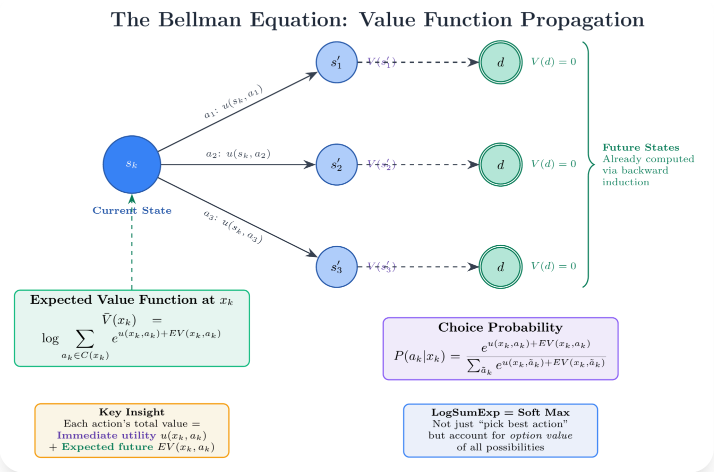
How It Works: Value Computation

THE CORE EQUATION

At each state, the agent chooses the action that maximizes:

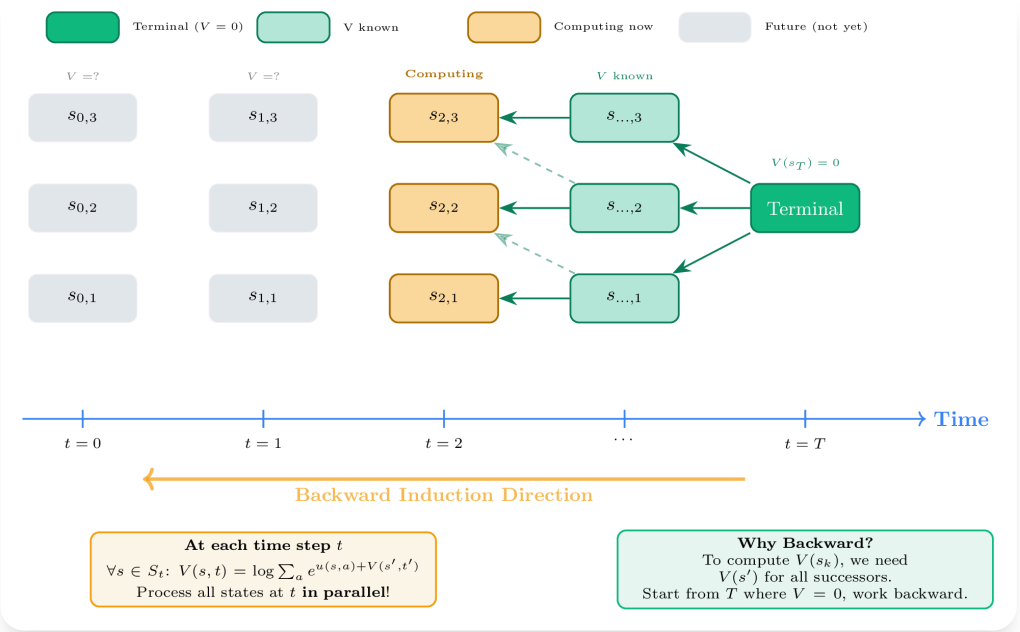
```
Immediate utility + Future utility
```

This is the Bellman equation from dynamic programming.



Key property: The "log-sum" formula gives both choice probabilities AND consumer surplus.

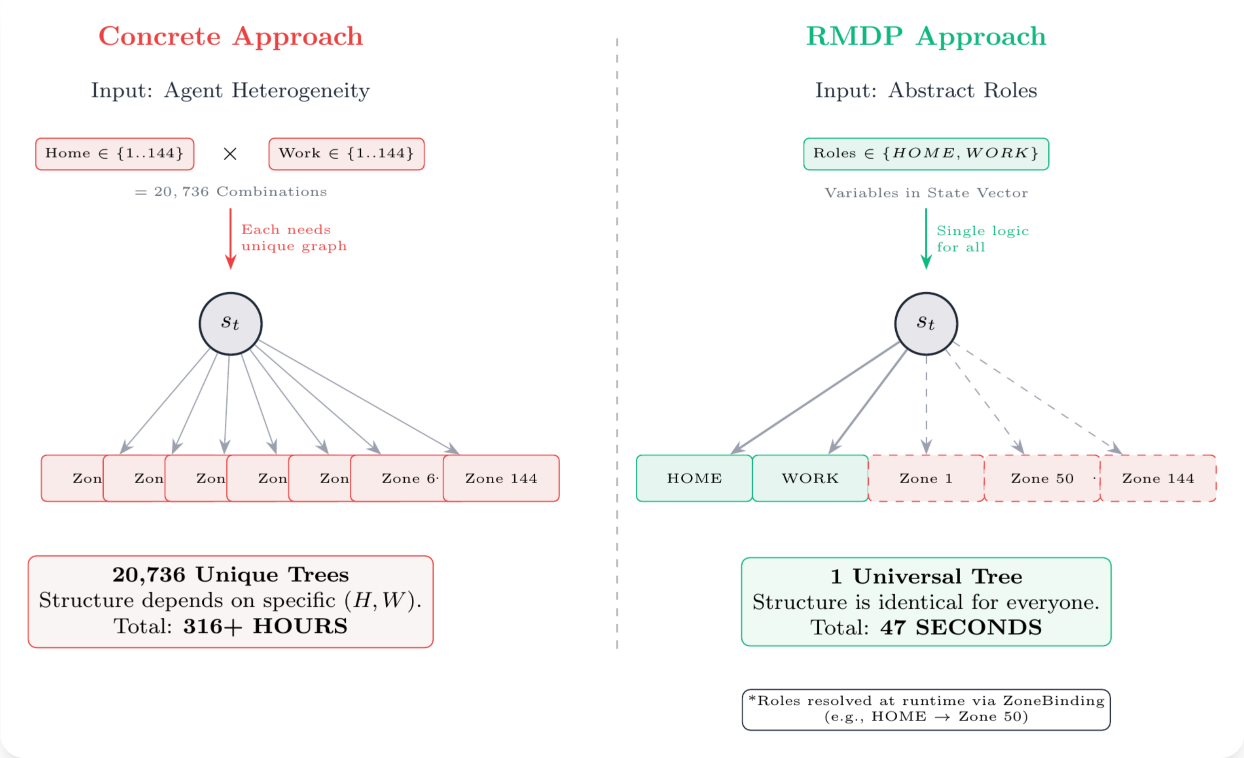
Working Backward in Time



Start at end of day (known value) → Work backward → Future values always available when needed

The Heterogeneity Challenge

Different agents have different home and work locations. Each (Home, Work) pair has a **unique graph structure.**



RMDP: The Key Idea

❌ Without RMDP

Different home/work → Different graph

- Alice (Zone 50 → Zone 73): Build graph #1
- Bob (Zone 12 → Zone 88): Build graph #2
- Carol (Zone 99 → Zone 25): Build graph #3
- ...

= 20,736 separate graphs!

✓ With RMDP

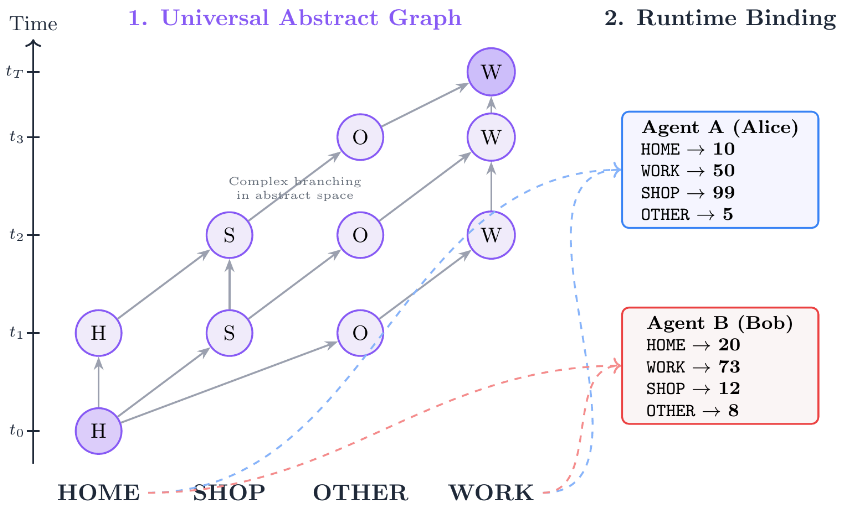
Use **abstract roles** instead of zones

- Graph uses: HOME, WORK, SHOP
- Alice binds: HOME=50, WORK=73
- Bob binds: HOME=12, WORK=88
- Same graph, different bindings!

= 1 universal graph!

Think of it like a map template: "Go from HOME to WORK" is the same instruction for everyone. Only the *addresses* differ.

RMDP: How Zone Binding Works



Same Graph, Different Reality:
When the graph says "Go to WORK", Alice goes to Zone 50, Bob goes to Zone 73.
This allows **millions of agents** to share a single graph structure in GPU memory.

Why This Works: Same Logic, Different Numbers

The Logic (Same for Everyone)

- "Leave home in the morning"
- "Travel to work"
- "Maybe stop for shopping on the way back"
- "Return home by evening"

The Numbers (Different for Everyone)

- Home → Work travel time: 25 min vs 40 min vs 15 min
- Which shops are nearby
- Specific zone preferences

RMDP separates WHAT decisions are made (graph structure) from WHERE they happen (zone binding)

Result: 316 hours → 47 seconds

5. Results

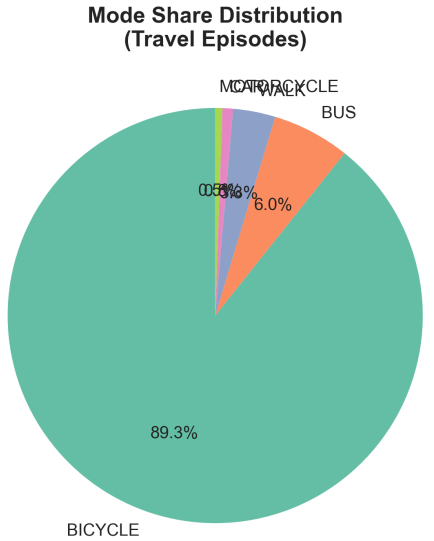
Performance and Validation

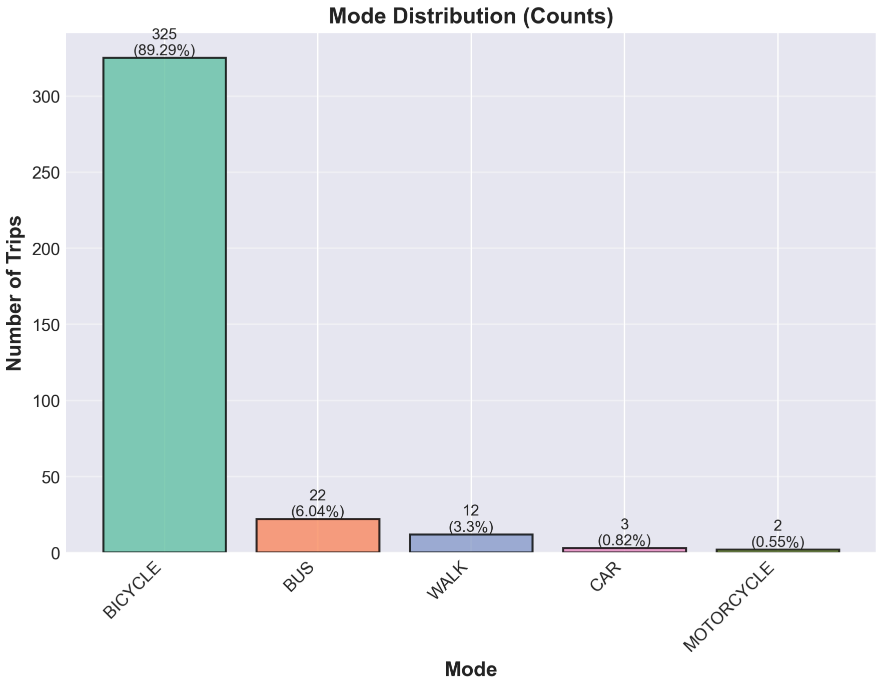
Performance Comparison

Approach	Time	Speedup
No Pruning (Baseline)	**71.1 hours**	1×
Dictionary-Based CPU	108s	2,370×
Tensor-Based CPU	65s	3,940×
GPU CUDA	8s	**32,000×**
+ Universal Graph	7s	**24.7×** vs per-home
+ RMDP (Final)	**8.7s**	**29,382×**

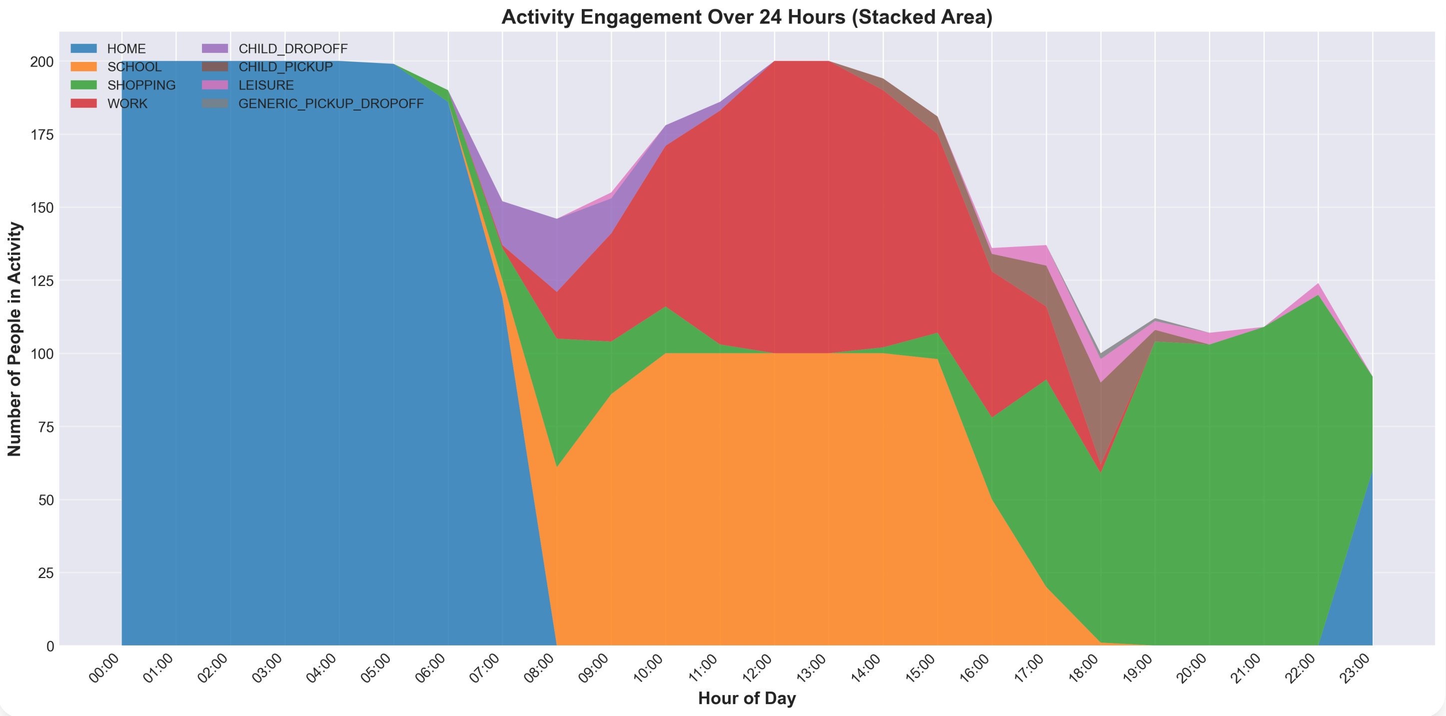
From 71 hours to 9 seconds — while maintaining optimality!

Simulation Results: Mode Distribution

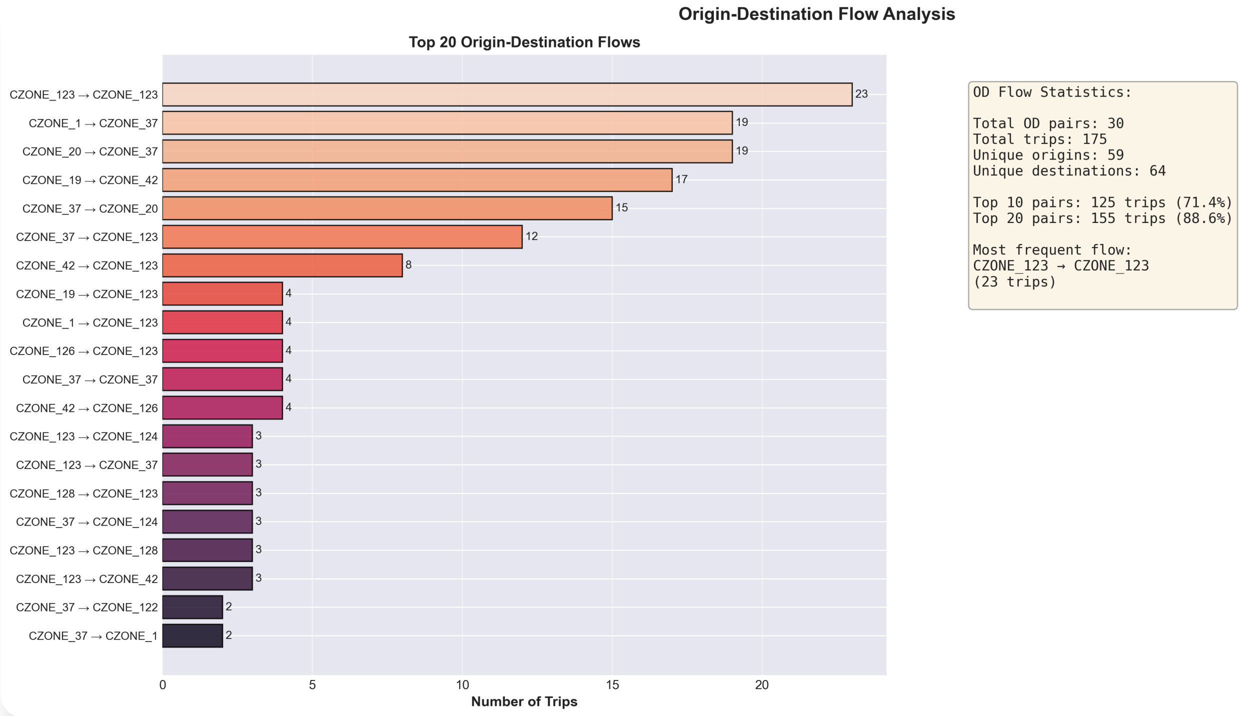




Simulation Results: Daily Activity Pattern



Simulation Results: Travel Flows



6. Conclusion

Contributions and Future Work

Research Contributions

1. INTEGRATION

First application of reachability analysis to discrete choice models

2. PERFORMANCE

29,382× speedup via RMDP + GPU

3. THEORETICAL BRIDGE

Space-Time Prism ≡ FRT ∩ BRT

4. PRACTICAL IMPACT

Enables population-scale simulation on consumer hardware

This work solves the simulation bottleneck, enabling estimation and traffic assignment integration.

Future Work: The Conditions Framework

Key Insight: Reachability analysis generalizes beyond simulation to handle *any constraint* expressible as "what states are reachable?"

Household Equity

Track travel burden within households

Gender equity under congestion pricing

Joint Activities

Unified abstraction for care activities

Childcare, eldercare, shared meals

Work Flexibility

Evaluate policy impacts

4-day week, flextime, remote work

Same computational foundation (this work) powers all applications via composable constraint specification.

Thank You

Solving the Curse of Dimensionality in DDCM

Graph-Based Computation | PyTorch & CUDA | CSR Graphs | Level-Sync BFS

29,382× speedup | 71 hours → 9 seconds | Optimal solutions preserved

Appendix

Technical Details (Backup for Q&A)

Appendix: The Bellman Equation

The value function can be defined recursively (Bellman 1957, Rust 1987)

$$\bar{V}(x_k) = \log \sum_{a_k \in C(x_k)} \exp\left[u(x_k, a_k) + EV(x_k, a_k)\right]$$

COMPONENTS

- **u(x,a)**: Immediate utility of action a in state x
- **EV(x,a)**: Expected future value after taking action a
- **log-sum-exp**: Aggregates over all possible actions

PROPERTIES

- Gumbel errors → Logit choice probabilities
- Log-sum = Consumer surplus
- Analytically differentiable

Appendix: MDP State-Action Space

STATE S_K

`t`	Time of day (96 steps)
`l`	Location (1,240 zones)
`p`	Current activity purpose
`τ`	Duration in activity
`m`	Mode (car/PT/walk/bike)
`h`	Mandatory activity history

ACTION A_K

$\tilde{d}$	Destination (1,240 zones)
$\tilde{m}$	Mode of transport
$\tilde{p}$	Activity purpose

`stay` = continue current activity.

Appendix: GPU Parallelization Details

Level-Synchronous BFS: Process all states at each time step in parallel

CSR: COMPRESSED SPARSE ROW

Dense Matrix	8.7 TB ❌
Edge List (COO)	2.6 GB
CSR	1.3 GB ✓

MEMORY BANDWIDTH

CPU (DDR4)	25 GB/s
GPU (GTX 1080 Ti)	484 GB/s

19× bandwidth advantage!

Appendix: Utility Function Components

Total Utility = Travel + Activity + Timing

$$u(s, a) = u_{\text{travel}} + u_{\text{start}} + u_{\text{act}} + \varepsilon(a)$$

u_{travel}

Disutility of travel: time, cost, mode constants

u_{start}

When/where to start: timing preferences, location attractiveness

u_{act}

Activity duration: marginal utility with diminishing returns

ε(a)

Random utility: i.i.d. Gumbel → Logit probabilities

Appendix: Approximation Methods

Basis Function BI

$V(s, home) \approx V_{ref}(s) + \beta \times \Delta travel\_time$

- Train on 8 representative zones
- Interpolate for remaining 136
- Accuracy: $R^2 = 0.982$
- Speedup: 9.3×

Multi-Fidelity BI

Full BI for some zones, interpolate rest

- Full BI for 15 representatives
- IDW interpolation for rest
- Accuracy: 99.6%
- Speedup: 2.5×

Appendix: Future Research Directions

Gap	Status	Enabled By
Simulation	✓ Solved (29,382×)	This work
Estimation	Ready to implement	Speedup
Traffic Assignment	Possible to integrate	Speedup
Correlation	Research needed	—
Households	Conditions Framework	Reachability
Long-term	Conditions Framework	Reachability

Future: The Conditions Framework generalizes reachability to handle household coordination, joint activities, and work flexibility policies.