A Computational Framework for Exact Activity-Based DDCM

Master Thesis Progress · April 2026

Azwan Nazamuddin
Graduate School of Innovation and Practice for Smart Society
Hiroshima University

Supervisor — Prof. Makoto Chikaraishi

Introduction

Why This Problem Matters

Why Do People Travel?

1970s	1990s	2000s+
Trip-Based	**Tour-Based**	**Activity-Based**
Where?	*How connected?*	*Why?*

Travel is **derived demand** — people travel *to do activities*, not for its own sake. Activity-based models capture **when, where, why, how** people travel, enabling welfare-consistent policy evaluation via log-sum accessibility (de Jong et al., 2007).

Bowman & Ben-Akiva (2001)

The Computational Challenge

"The number of travel patterns could easily exceed the ***number of atoms in the universe****."* — Västberg (2018)

Why so large?

State multiplies across dimensions:

`time × zone × activity × duration × mode × vehicle × history`

≈ **145 million states** for one city.

The practical cost

Västberg et al. (2020): ~10 s per agent for likelihood evaluation.

- **100K agents** → ~1,000 CPU-days
- **Memory** → 6.7 TB Q-table

Exact city-scale estimation — practically impossible.

DDCM: The Right Framework

Computational Process Models

Rule/heuristic-based. STARCHILD, ALBATROSS.

- ✓ Realistic scheduling
- ✗ No welfare / consumer surplus

Random Utility Models (MEV)

Utility-maximizing. MNL, Nested Logit.

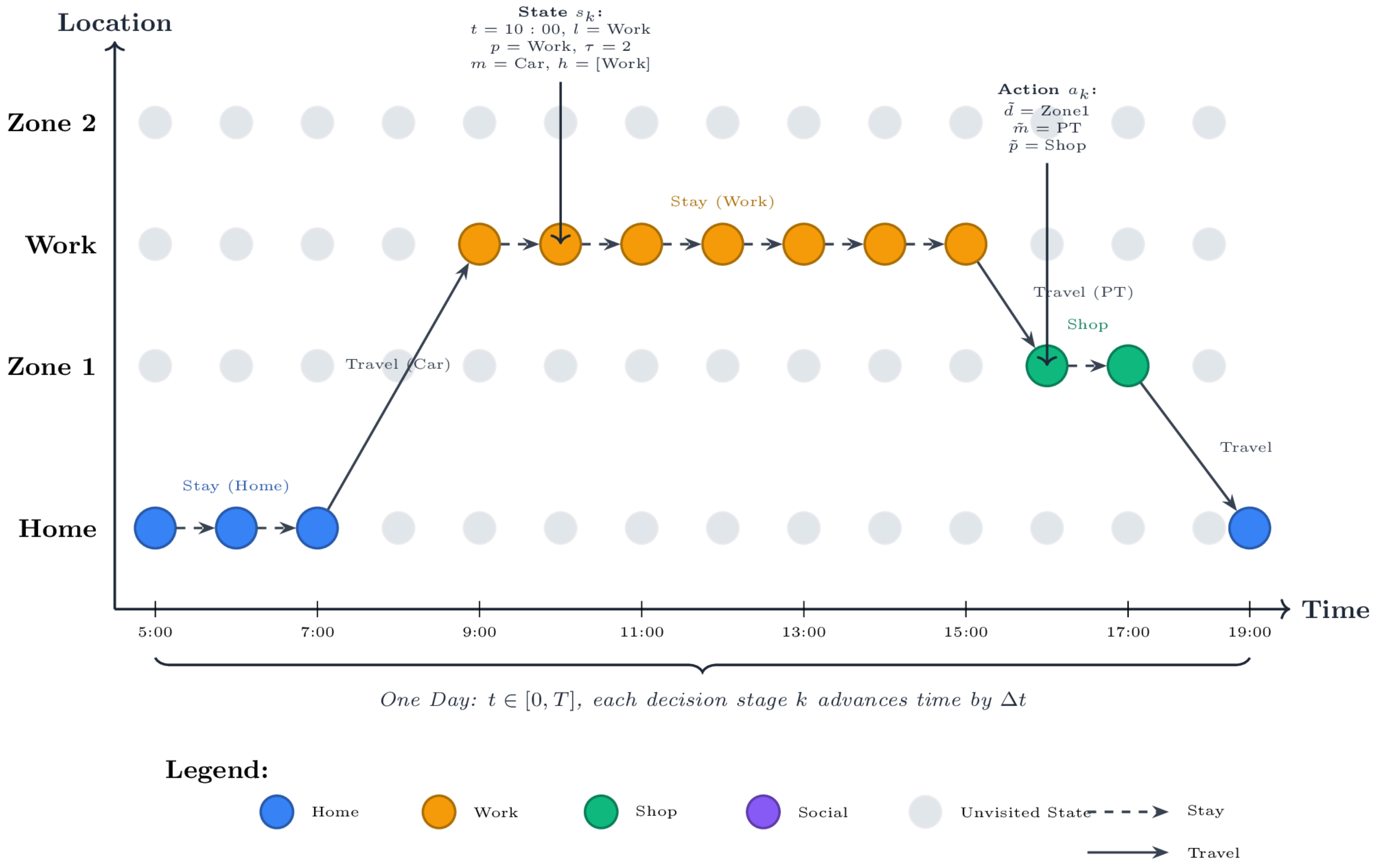
- ✓ Log-sum welfare
- ✗ Weak intertemporal treatment

DDCM (Västberg et al., 2020) — RUM + sequential MDP. Each state s has a value function:

$$\bar{V}(s) = \log \sum_{a \in \mathcal{C}(s)} \exp\big[u(s,a) + \bar{V}(s')\big]$$

The **log-sum is consumer surplus** — welfare-consistent (de Jong et al., 2007). Backward induction from T to 0.

The Activity-Travel MDP



At each 15-min step: choose destination, mode, and activity (Västberg et al., 2020)

Our Framework

DAG Structure · Reachability Pruning · $\mu(t)$ Utility — GPU-parallelized via graph traversal

Framework Overview

1. DDCM as a DAG

Time only moves forward → no cycles. Finite-horizon DDCM is a Directed Acyclic Graph. Agents of the same activity-sequence type share one graph; individual constraints as masks.

2. Reachability Pruning

Forward BFS with space-time prism constraints. **145M → 1.5M nodes (~1%)**. Exact optimality preserved.

3. μ(t) Utility Profile

Time-varying marginal utility profiles. Activity timing and sequencing emerge from preference gradients — no hard-coded rules.

The DAG structure enables **GPU-parallel backward induction** — each level is processed in one batched kernel, with individual constraints applied as masks.

Contribution 1 — DDCM as a DAG

Key Structural Insight

Time only moves forward → the state-transition graph has **no cycles**. Backward induction on a finite-horizon DDCM is exactly a reverse topological traversal over a Directed Acyclic Graph.

Bellman (1957); Rust (1987); Dudzik & Veličković (2022)

Bellman Structure → Graph

- States → **Vertices**
- Actions → **Edges**
- Forward BFS → reachable nodes
- Backward BI → reverse traversal

Why This Matters

- DAG levels are independent → GPU processes each level in one batched kernel
- Only reachable nodes needed → motivates Contribution 2
- Same-type agents share one graph; individual constraints applied as **masks**

Contribution 2 — Reachability-Based Pruning

The Problem

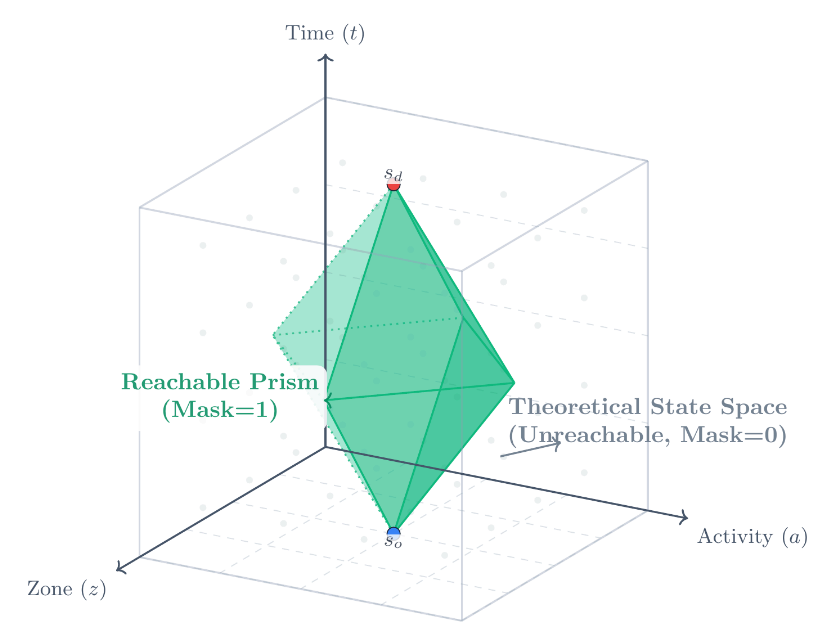
Each DAG has ~145M nodes. Most are physically unreachable given travel times and space-time constraints.

Forward BFS with Space-Time Prism

Level-synchronous BFS (Berrendorf et al., 2014) from feasible initial states, embedding prism constraints (Hägerstrand, 1970; Oyama & Hato, 2019). One forward pass, stored as CSR — exact, no approximation.

145M → 1.5M nodes (~1%)

Exact optimality — same value functions as the full graph.



New Utility Specification

$\mu(t)$: Activity Timing from Preference Gradients

Why a New Utility Specification?

Västberg et al. (2020)

- Schedule-delay penalties + **hard time windows** (e.g. work starts 6 am–10 pm)
- Window boundaries are inputs, not outcomes
- **Timing partly constrained by rules**

Our Goal: Fully Emergent Timing

- No hard windows. Each activity has a **profile** $\mu_a(t)$ — piecewise or Gaussian-mixture
- Timing and trip-making emerge from profile gradients alone
- **Behavioral patterns are outputs, not inputs**

Grounded in the temporal utility profile framework of **Supernak (1992)** and **Joh et al. (2003)**.

$\mu(t)$: Activity and Travel Utility

Core idea (Supernak, 1992; Joh et al., 2003) — activity a becomes the modal choice when its net value $\mu_a(t) \cdot \Delta t - \text{travel} - 2|c_{\text{change}}| + \bar{V}(\text{dest})$ exceeds the home alternative. Under the logit kernel, every feasible action retains positive probability; the utility gap controls the ratio of choice probabilities.

Home

μ_{home}: flat constant. Every trip must clear this floor.

Work / School

Piecewise around (t_s, t_e):

$$\mu_{\text{work}}(t) = \begin{cases} \delta - \alpha(t_s - t) & t < t_s \\ \delta & t_s \le t \le t_e \\ \delta - \beta(t - t_e) & t > t_e \end{cases}$$

Shop / Leisure

Business-hour profile:

$$\mu_{\text{shop}}(t) = \beta_1 P_{\text{open}}(z, t) + \beta_0$$

P_{open}: Google Maps POI, Gaussian-mixture fit.

Travel

Scales MNL mode utility:

$$u_{\text{travel}} = \theta_{\text{travel}} \cdot v_m(l, d, t)$$

θ_{travel} aligns scales with activity utility.

Full one-stage utility for a trip $(l \to d, m)$ + activity a:

$$u = \mu_a(t) \cdot \Delta t + \theta_{\text{travel}} \cdot v_m(l, d, t)$$

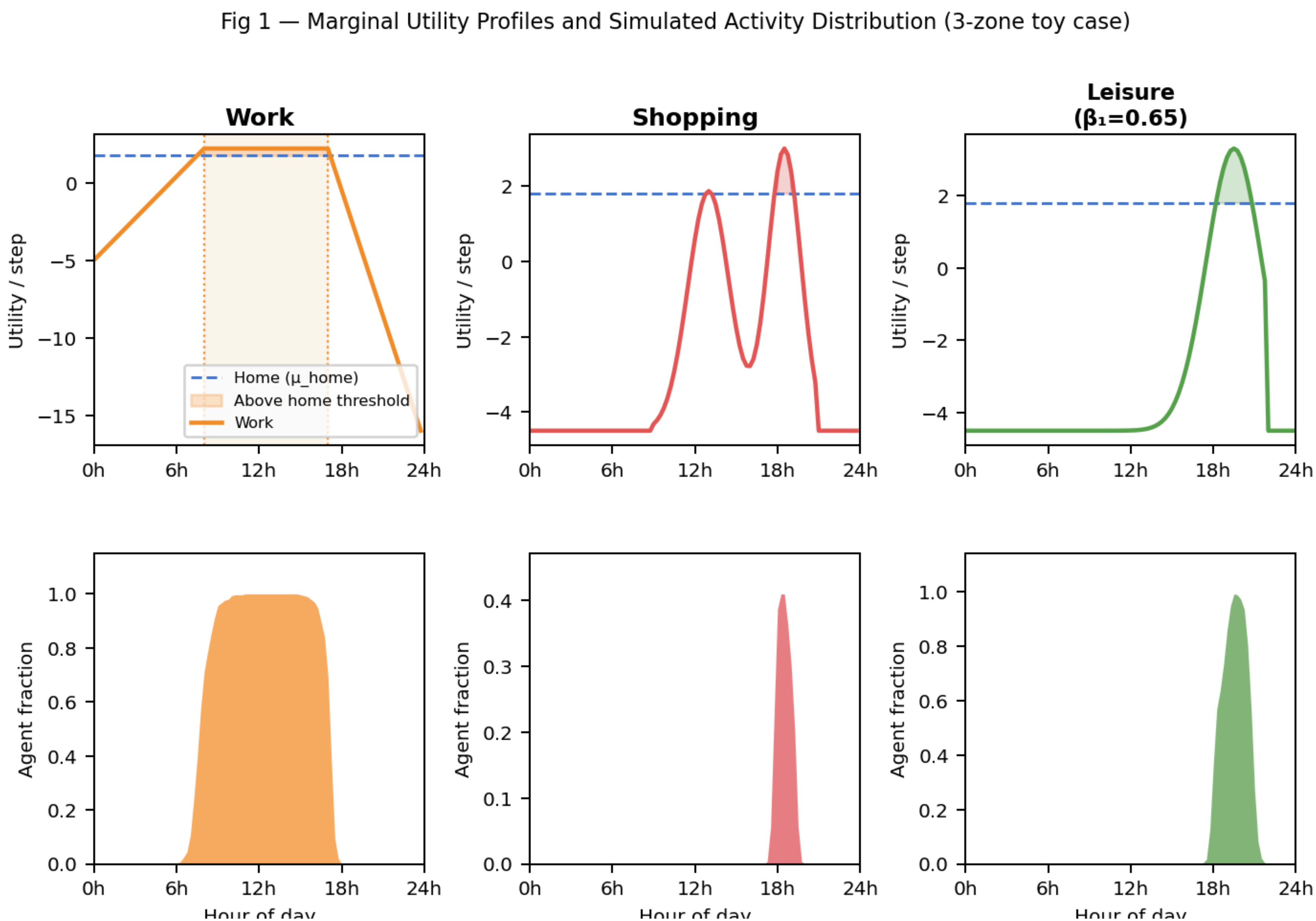
Symbol	Description	Activity
δ	On-schedule marginal utility (per min)	Work
α	Earliness penalty rate	Work
β	Lateness penalty rate	Work
$\beta_{1,\text{shop}}$	Shopping $\times P_{\text{open}}$ sensitivity	Shopping
$\beta_{0,\text{shop}}$	Shopping base marginal utility	Shopping
$\beta_{1,\text{leis}}$	Leisure $\times P_{\text{open}}$ sensitivity	Leisure
$\beta_{0,\text{leis}}$	Leisure base marginal utility	Leisure
c_{change}	Activity-switching cost	All
μ_{home}	Home flat marginal utility (per min)	Home
θ_{travel}	Travel disutility scale	Travel

10 parameters across 3 profile types

- **Work** (δ, α, β): piecewise schedule shape. t_s, t_e are diary inputs.
- **Shop / Leisure** (β_1, β_0): sensitivity to business-hour profile.
- **Home** (μ_{home}): reservation floor.
- **Switching** (c_{change}): per-change cost at departure + arrival.
- **Travel** (θ_{travel}): MNL ↔ DDCM scale.

The "cost of moving": c_{change} and μ_{home} jointly anchor trip frequency — each trip must clear μ_{home} and pay $2c_{\text{change}}$. Work/shop/home timing is **endogenous**.

Activity Patterns — Simulated Utility Profiles



Activity Patterns — Behavioral Interpretation

Work

Departs when profile exceeds the home floor. Morning peak emerges from δ and diary window (t_s, t_e).

Shopping

Trips cluster at business-hour peaks of $P_{\text{open}}(t)$ — a utility signal, not a constraint.

Leisure

Active only when it clears μ_{home}. Fig 1 raises $\beta_{1,\text{leis}}$ to 0.65 for illustration; at baseline (0.4) it stays below the floor.

All 5 sanity checks pass: home at night · work dominates daytime · correct trip counts · work-start window · mode shares.

Results

Computational · Behavioral · Estimation

Computational Results

Metric	Conventional	Proposed	Improvement
States processed	145 million	1.5 million	99% reduction
Full pipeline time	~69 hours (est.)	105 seconds	~2,400×
Memory (Q-table / CSR)	6.7 TB	6.5 GB	~1,000×
Simulation (1,000 agents)	—	3.9 seconds	—
Per-agent cost	121 ms	3.9 ms	~31×

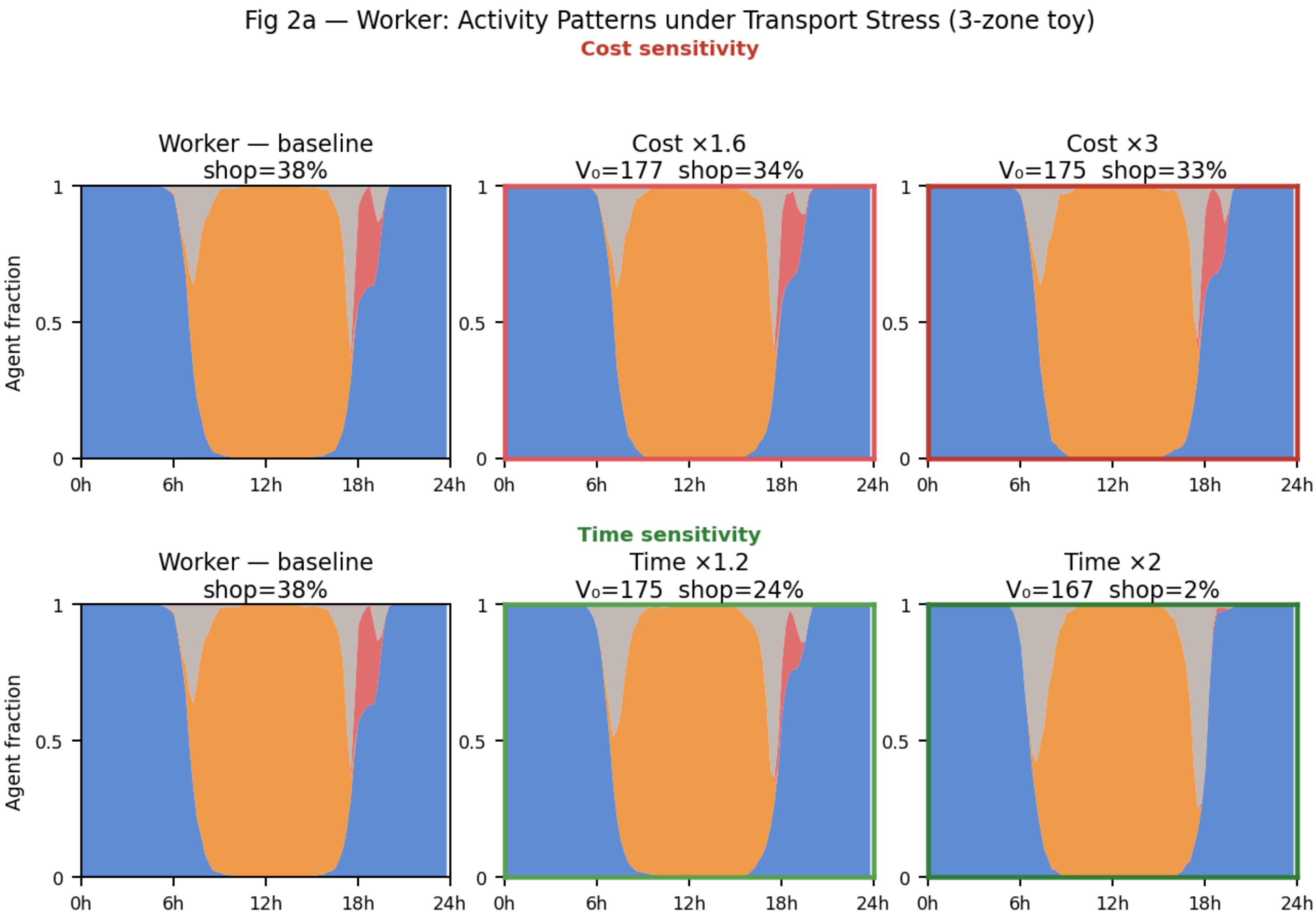
Exact optimality preserved

The pruned graph contains all states on any feasible trajectory. Identical value functions to the full 145M-state graph.

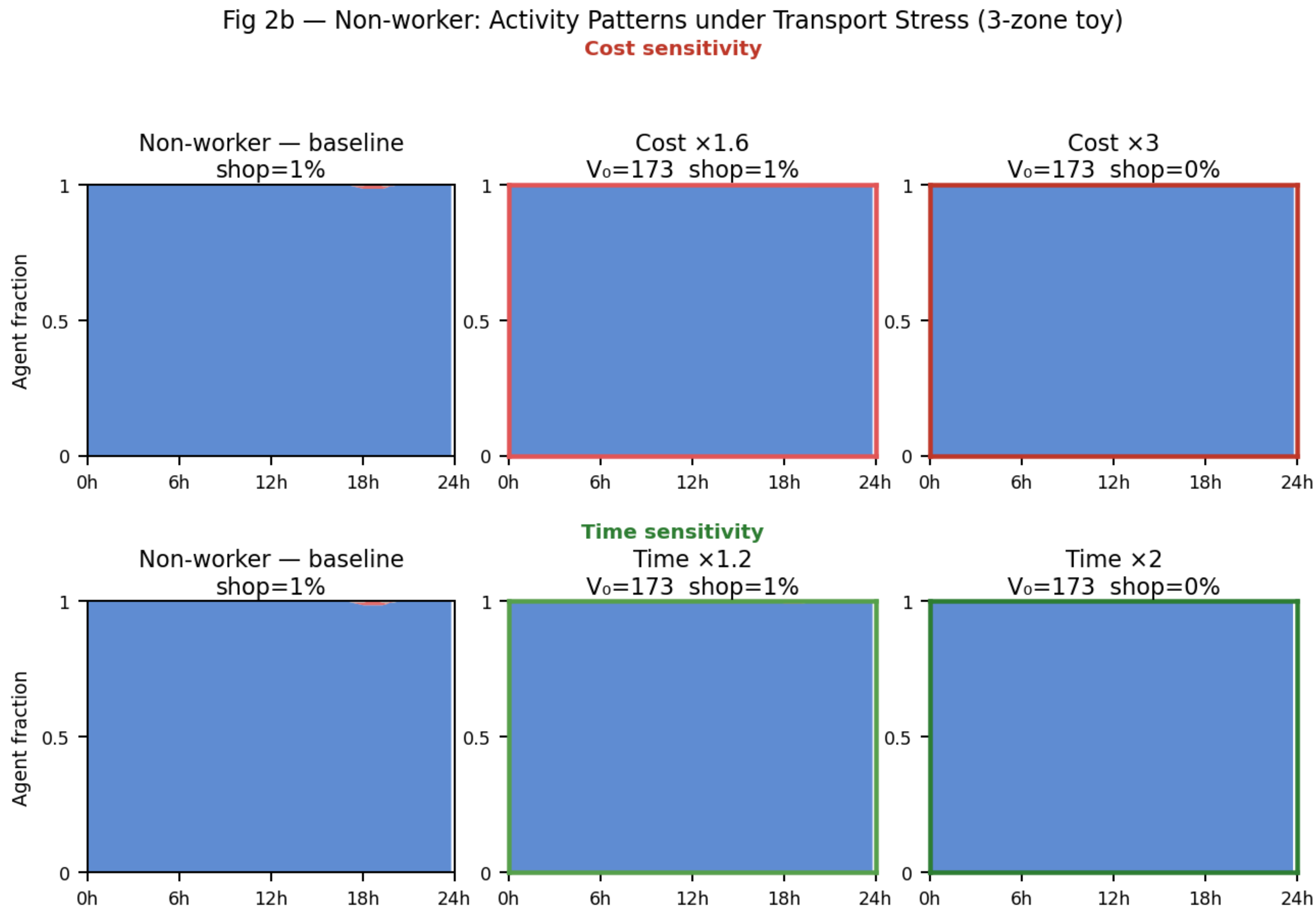
What the 69h estimate is

BI over the unpruned 145M-state Cartesian product. The 105s figure includes BFS + pruned BI on the CSR graph.

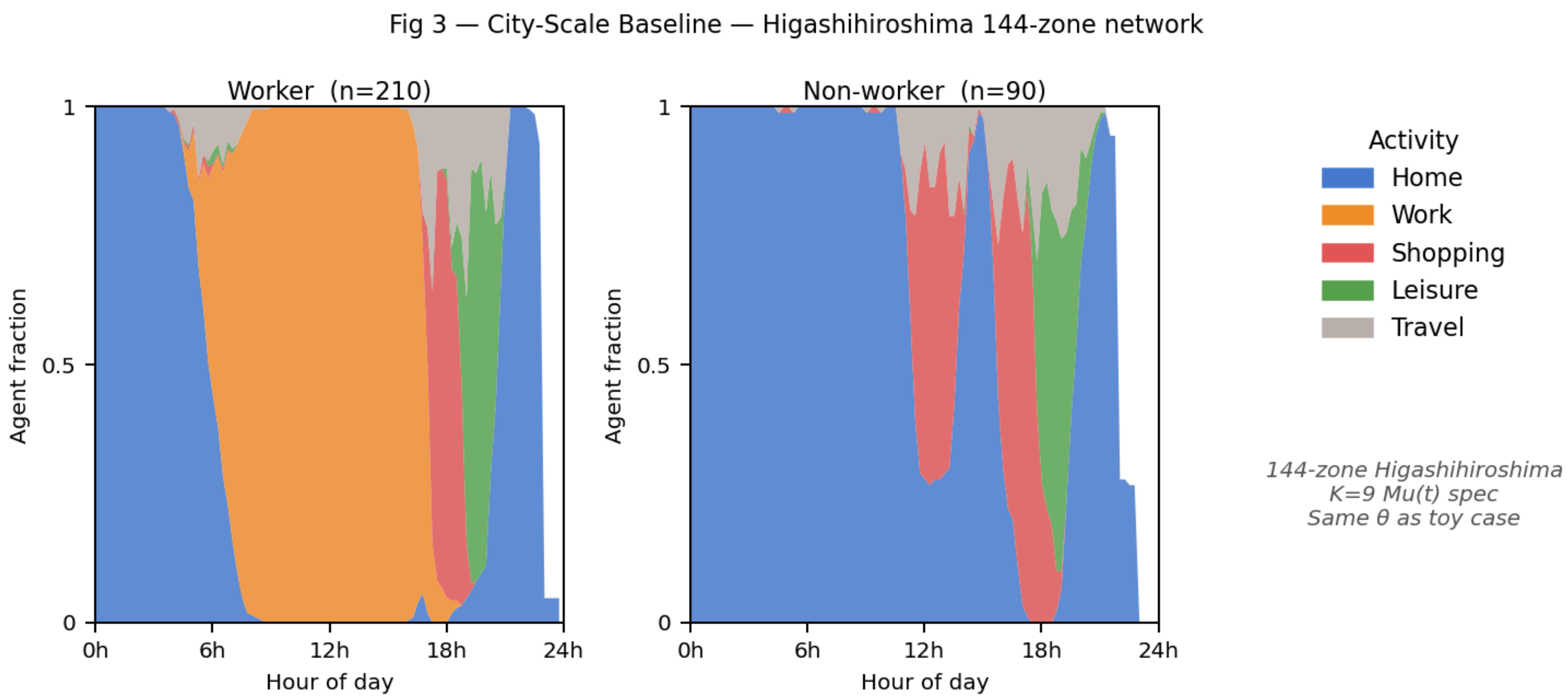
Toy Case: Worker Transport Sensitivity



Toy Case: Non-Worker Transport Sensitivity



City-Scale Simulation



144 zones · Higashi-Hiroshima, Japan · 4 activity-sequence groups cover the full population · full city simulation in under 3 minutes

Estimation: NFXP Pipeline

Log-likelihood over N agents, K_n steps:

$$\ell(\theta) = \sum_n \sum_k \log P(a_{k,n} \mid s_{k,n};\, \theta)$$

$$P(a \mid s;\, \theta) = \exp\big[u(s, a; \theta) + \bar{V}(s'; \theta) - \bar{V}(s; \theta)\big]$$

Analytical gradient via a second BI pass — exact, no finite differences (Fosgerau et al., 2013; Baydin et al., 2018):

$$\frac{\partial \ell}{\partial \theta_q} = \sum_{n,k} \left[x^q_{k+1|k} - \frac{\partial \bar{V}(s_{k,n})}{\partial \theta_q} \right], \qquad \frac{\partial \bar{V}}{\partial \theta_q} \text{ solved by the same Bellman recursion.}$$

NFXP: Rust (1987); Västberg et al. (2020)

Outer — BFGS

Maximizes $\ell(\theta)$ over K=10. Target $\|\nabla \ell\|_\infty < 0.001$.

Inner — Backward Induction

Per BFGS step: BI on pruned DAG $\rightarrow \bar{V} \rightarrow$ simulate $\rightarrow \ell$.

Gradient — Second BI Pass

$\nabla_\theta \bar{V}$ propagated backward. Same cost order as inner BI.

~22 min/iter · 14 groups · ~5.9M states/group · ~7 GB GPU.

Estimation: Current Results

1,368 workers · 29 groups · bounded $c_{\text{change}} \in (-2.5, 0)$ · best checkpoint at BFGS iter 19

Parameter	Description	$\hat{\theta}$
δ	On-schedule utility (per min)	0.0266
α	Earliness penalty rate	0.00102
β	Lateness penalty rate	0.00300
$\beta_{1,\text{shop}}$	Shop P_{open} sensitivity	0.281
$\beta_{0,\text{shop}}$	Shop base utility	−0.871

Parameter	Description	$\hat{\theta}$
$\beta_{1,\text{leis}}$	Leisure P_{open} sensitivity	0.353
$\beta_{0,\text{leis}}$	Leisure base utility	−0.162
c_{change}	Activity-switching cost	**−2.500** ⚠ bound
μ_{home}	Home floor utility	0.102
θ_{travel}	Travel disutility scale	2.00

Best $\ell = -28{,}708.6 \cdot \|\nabla \ell\|_\infty = 0.78$ (threshold 0.001) · **partial convergence —** BFGS terminated at iter 35 with precision-loss message; c_{change} pinned at the lower bound. δ, μ_{home} now carry the expected positive sign (vs. previous SRS200 run).

Estimation: Convergence Analysis

What we observed

Bounding $c_{\text{change}} \in (-2.5, 0)$ *did* keep BFGS out of the deep basin at -3.86, but the optimizer still drove it to the lower bound and stalled. Activity params barely moved (13 plateau iters at $\ell \approx -28{,}709$); θ_{travel} ran off to 2.0.

Suspected: shallow ridge along c_{change} direction — activity utilities weakly identified since the data's switching-frequency signal is absorbed by c_{change} wherever it sits. *Identification issue, not optimizer issue.* **Under investigation.**

Brief recommendation

1. **Verify formulation first.** c_{change} enters twice per trip; check the score is not monotone by construction before treating this as unidentified.
2. **Profile-likelihood sweep** over $c_{\text{change}} \in \{-2.5, \ldots, 0\}$ — visualises the ridge.
3. **Fallback:** fix c_{change} at the MNL switching-cost prior (-0.3) and estimate the remaining 9 params — yields a paper-ready table.

What's Next

1. Converge the estimation

Re-estimation running. Resolve c_{change} basin via multi-start BFGS with constrained init. (NOT FIXED)

2. Welfare analysis

Log-sum consumer surplus across transport scenarios → city-scale CBA.

3. μ(t) theory

Alternative profile shapes, richer shopping/leisure specs, sensitivity analysis.

4. Complete the paper

Full write-up with converged estimates + city-scale sensitivity + welfare.

5. Broader PhD direction

DDC-Bridge: DDCM ↔ DL/RL via MaxEnt IRL ≡ logit DDC (Ermon, 2015) and BI ≡ log-semiring layered networks (Dudzik & Veličković, 2022).