E-COMPANION FOR

Dynamic Scheduling of Recurring Multi-Session Appointments with Heterogeneous Clients

Appendix A: Proofs for Section 4.2

A.1. Proof of Lemma 1

Given the slot-selection guideline in Section 4.1, the acceptance decision is governed by the reduced Bellman equation (8). Under this representation, it is optimal to reject a request (L, J) in schedule state Y if:

$$V(f(Y,0,L)) - V(f(Y,\hat{j},L)) \geq r,$$

where $\hat{j}$ is defined in (7). If $V(f(Y,\hat{j},L))$ is non-increasing in L, then this inequality implies:

$$V(f(Y,0,L+1)) - V(f(Y,\hat{j},L+1)) = V(f(Y,0,L)) - V(f(Y,\hat{j},L+1)) \geq r.$$

Therefore, if it is optimal to reject a request of duration L, it is also optimal to reject a request of duration $L+1$ in the same schedule state. □

A.2. Proof of Lemma 2

For the purpose of this structural argument, we temporarily introduce explicit time indices $t, t+1, \ldots$; these indices do not appear in the time-homogeneous MDP formulation but are useful for comparing future action sequences.

Lemma 2 reflects the intuitive idea that the value of a schedule should not increase when resources are occupied for more periods. Fix a state Y^{t-1} and a slot $j \in \mathcal{J}$, and consider the two successor states

$$Y^{t,(L+1)} := f(Y^{t-1}, j, L+1), \qquad Y^{t,(L)} := f(Y^{t-1}, j, L).$$

Let $\{a^{t+i}\}_{i=0}^{\infty}$ denote an optimal action sequence beginning at the state $Y^{t,(L+1)}$. Applying this same sequence to the alternative state $Y^{t,(L)}$ yields a feasible (but possibly suboptimal) policy τ for $Y^{t,(L)}$. Therefore,

$$V^{\tau}\big(Y^{t,(L)}\big) \;\leq\; V\big(Y^{t,(L)}\big).$$

Moreover, since $Y^{t,(L)}$ and $Y^{t,(L+1)}$ differ only by one additional unit of remaining occupancy in slot j, applying the same sequence $\{a^{t+i}\}_{i\geq 0}$ produces identical assignments, rejections, and rewards from period t onward. Hence,

$$V^{\tau}\big(Y^{t,(L)}\big) \;=\; V\big(Y^{t,(L+1)}\big).$$

Combining the two displays yields

$$V\big(Y^{t,(L+1)}\big) \le V\big(Y^{t,(L)}\big),$$

which completes the proof. □

A.3. Proof of Lemma 3

Proof: Conditional on the slot-selection guideline defined in Section 4.1, the acceptance decision follows the reduced Bellman equation (8). Under this equation, it is optimal to accept a request (L, J) in schedule state Y if:

$$V(f(Y,0,L)) - V(f(Y,\hat{j},L)) \le R(L),$$

where $\hat{j}$ is defined in (7).

We begin by expressing the left-hand side of the inequality as a telescoping sum:

$$V(f(Y,0,L)) - V(f(Y,\hat{j},L)) = \sum_{h=0}^{L-1} \Big[V(f(Y,\hat{j},h)) - V(f(Y,\hat{j},h+1))\Big].$$

Since $V(f(Y,j,L))$ is non-increasing and convex in L, the sequence of differences is non-decreasing in L, i.e.,

$$V(f(Y,j,L-1)) - V(f(Y,j,L)) \le V(f(Y,j,L-2)) - V(f(Y,j,L-1)).$$

Therefore, for any L, the sum above satisfies:

$$L \cdot \Big[V(f(Y,\hat{j},L-1)) - V(f(Y,\hat{j},L))\Big] \le \sum_{h=0}^{L-1} \Big[V(f(Y,\hat{j},h)) - V(f(Y,\hat{j},h+1))\Big], \tag{EC.1}$$

Since $V(f(Y,\hat{j},L))$ is discretely convex in L for any fixed Y and $\hat{j}$, the sequence of forward differences $\Delta_L = V(f(Y,\hat{j},L)) - V(f(Y,\hat{j},L+1))$ is non-decreasing, so the right-hand side of (EC.1) upper-bounds the linear extrapolation on the left, and the acceptance condition propagates from L to $L+1$. Thus if it is optimal to accept the request at duration L, we have:

$$L \cdot \Big[V(f(Y,\hat{j},L-1)) - V(f(Y,\hat{j},L))\Big] \le R(L).$$

Now, using the convexity of $V(f(Y,\hat{j},L))$, we have:

$$V(f(Y,\hat{j},L)) - V(f(Y,\hat{j},L+1)) \le V(f(Y,\hat{j},L-1)) - V(f(Y,\hat{j},L)) \le \frac{R(L)}{L}.$$

Also, since $R(L)$ is increasing and convex, we obtain:

$$R(L) + \frac{R(L)}{L} \le R(L+1).$$

Combining all the above, we conclude:

$$\begin{aligned} V(f(Y,0,L)) - V(f(Y,\hat{j},L+1)) &= \Big[V(f(Y,0,L)) - V(f(Y,\hat{j},L))\Big] \\ &\quad + \Big[V(f(Y,\hat{j},L)) - V(f(Y,\hat{j},L+1))\Big] \\ &\le R(L) + \frac{R(L)}{L} \le R(L+1), \end{aligned}$$

which implies that the acceptance condition is satisfied at $L+1$ as well. Therefore, if it is optimal to accept a request of duration L, it is also optimal to accept a request of duration $L+1$ in the same schedule state. □

A.4. Proof of Lemma 4

Monotonicity. This property follows directly from Lemma 2, which established that $V(f(Y,j,L))$ is non-increasing in L for all $Y \in \mathcal{Y}$ and $j \in \mathcal{J}$. We therefore focus on proving convexity.

Convexity (in discrete L).

We prove convexity by value iteration and an operator-preservation argument, using *discrete convexity*: a function $g : \mathbb{Z}_+ \to \mathbb{R}$ is convex if its forward differences are non-decreasing, i.e.,

$$\Delta g(L) := g(L{+}1) - g(L) \quad \text{satisfies} \quad \Delta g(L{+}1) \ge \Delta g(L) \text{ for all feasible } L.$$

Let $V_n(Y) \xrightarrow[n\to\infty]{} V(Y)$ denote the value-iteration sequence. For fixed Y, j, define the one-dimensional slice

$$W_n(L) := V_n(f(Y,j,L)), \qquad L \in \mathcal{L}.$$

We will show: if W_n is discrete-convex in L, then so is W_{n+1}. Since $W_0 \equiv 0$ is convex, the limit $W = \lim_n W_n$ is convex as a pointwise limit of convex functions.

We use two facts:

1. **Piecewise-affine dependence of the next state on L.** From Equation (1), $f(Y,j,L) = \max(Y{-}1,0) + Le_j$. For fixed Y, this is affine in L (indeed linear) in the j-coordinate and independent of L in the others once $\max(Y{-}1,0)$ is fixed. Hence the map $L \mapsto f(Y,j,L)$ is componentwise non-decreasing and piecewise affine in L.
2. **Max and expectation preserve convexity.** For functions of L, the pointwise maximum of discrete-convex functions is discrete-convex, and taking expectations over variables other than L preserves discrete convexity (Bertsekas (2012)).

Consider the *Bellman update*. Let (H, J) denote the next-request variables (duration and preference set) with distribution $p_{H,J}$ (Section 3.2). From Equation (3), the Bellman update at state X is

$$V_{n+1}(X) = \mathbb{E}_{(H,J)}\Big[\max\Big\{\gamma V_n(f(X,0,H)), \max_{m \in \widetilde{J}(X,J)} (R(H) + \gamma V_n(f(X,m,H)))\Big\}\Big].$$

Set $X_L := f(Y, j, L)$. Then

$$W_{n+1}(L) = V_{n+1}(X_L) = \mathbb{E}_{(H,J)}\left[\max\left\{\gamma V_n(f(X_L, 0, H)), \max_{m \in \widetilde{J}(X_L, J)} \left(R(H) + \gamma V_n(f(X_L, m, H))\right)\right\}\right].$$

For fixed (H, J) and action $a \in \{0\} \cup \mathcal{J}$, the map

$$L \longmapsto V_n(f(X_L, a, H)) = V_n(f(f(Y, j, L), a, H))$$

is a composition of V_n with the state mapping $L \mapsto f(f(Y, j, L), a, H)$. The latter is componentwise non-decreasing and piecewise affine in L by (1), hence preserves discrete convexity of W_n. Multiplying by $\gamma \in (0, 1)$ and adding the constant $R(H)$ (independent of L) preserve discrete convexity. Taking a pointwise maximum over a preserves discrete convexity, and taking expectation over (H, J) (which does not involve L) also preserves discrete convexity. Therefore W_{n+1} is discrete-convex whenever W_n is.

Finally, since $W_0 \equiv 0$ is convex, induction implies W_n is convex for all n, and pointwise convergence of V_n to V ensures that $W(L) = \lim_n W_n(L)$ is discrete-convex as well. This proves convexity of $L \mapsto V(f(Y, j, L))$.

Combining monotonicity and convexity completes the proof.

Remark EC.1 (Variable roles)**.** *Convexity is taken with respect to the current request's duration L that determines the next state $f(Y, j, L)$. The maximization and expectation operators in the Bellman update are over future request variables (H, J), which are random and independent of the current L. Making this separation explicit clarifies why the Bellman operator preserves convexity in L.*

Appendix B: Heuristic Threshold Definition and Calibration

To implement the Traffic Light heuristic, it is necessary to define the thresholds that determine whether a request is accepted at a given occupancy state. The procedure consists of two main steps: (i) classifying the current schedule state into one of three occupancy categories (green, orange, or red) and (ii) selecting the optimal threshold values through simulation-based tuning.

B.1. Occupancy Rate Measure

At each decision point, we evaluate the schedule over a planning horizon of T future periods and compute the occupancy rate:

$$\text{Occupancy Rate} = \frac{\text{Number of occupied slots in the next } T \text{ periods}}{\text{Total number of slots in the next } T \text{ periods}}.$$

For example, consider a schedule with $|\mathbb{J}| = 10$ slots and state vector $Y = (4, 0, 2, 0, 5, 0, 6, 0, 0, 8)$, where each entry indicates how many periods into the future the slot is occupied. Setting $T = 4$

means we consider 40 future slot-periods, of which 18 are occupied (two from slot 3 and four from each of the other occupied slots), yielding an occupancy rate of 0.45. In our experiments, we set T to be (max treatment length − min treatment length)/2, which provides a balanced look-ahead horizon relative to the range of possible treatment durations. Then, two boundaries values are defined:

- Low-occupancy bound (LOB): if the occupancy rate is below LOB, the state is green.
- Moderate-occupancy bound (MOB): if the occupancy rate is above MOB, the state is red.
- Otherwise, the state is orange.

To determine the best-performing thresholds, we conducted an exhaustive search over the set:

$$LOB \in \{0.2, 0.3\}, \quad MOB \in \{0.5, 0.6, 0.7, 0.8\}.$$

For each (LOB, MOB) pair, we simulated the heuristic under all acceptance threshold combinations consistent with the decision logic outlined in Section 5, separately for client-dominance and provider-dominance scenarios. The performance of each configuration was measured by the average objective value over the 20 randomly generated instances per experimental condition. Following this search, we fixed $LOB = 0.3$ and $MOB = 0.6$ for the main experiments, as this combination consistently performed well across the tested instances. More importantly, the heuristic exhibited relatively stable performance across a range of nearby occupancy boundaries, indicating that the results are not driven by fine parameter tuning. Performance degradation became noticeable mainly when the occupancy thresholds generated highly unbalanced state partitions. For example, boundary combinations such as $(0.2, 0.8)$ tend to produce very few states classified as either low- or high-occupancy, while concentrating most schedule states in the middle category, thereby reducing the discriminatory power of the aggregation. The model's performance, however, was not highly sensitive to variations within the range $LOB \in \{0.2, 0.3\}$ combined with $MOB \in \{0.6, 0.7\}$.

B.2. Acceptance Thresholds by Scenario

The following tables present the acceptance thresholds (TR) obtained for the different experimental settings. Table EC.1 reports the results for the Client-Dominance (CD) case, Table EC.2 for the Provider-Dominance (PD) case with a linear revenue function $R(L) = L$, and Table EC.3 for the PD case with a strictly convex revenue function $R(L) = L^2/100$.

The tables above report the treatment length thresholds for accepting a request under different occupancy states and slot types. In the Client-Dominance setting, a request is accepted if its length L is no greater than TR, so lower values of TR relative to the maximal possible treatment length correspond to stricter acceptance policies. In the Provider-Dominance settings, by contrast, a request is accepted if L is at least TR, which means that higher values of TR relative to the

Table EC.1 Acceptance thresholds for Client-Dominance (CD) scenarios

Scenario	TR(O,P)	TR(R,P)	TR(O,NP)	TR(R,NP)
WL	25 (83%)	20 (67%)	30 (100%)	30 (100%)
WM	35 (70%)	30 (60%)	50 (100%)	45 (90%)
WH	50 (50%)	20 (20%)	100 (100%)	45 (45%)
EL	N/A	N/A	30 (100%)	30 (100%)
EM	N/A	N/A	45 (90%)	45 (90%)
EH	N/A	N/A	60 (60%)	60 (60%)

Table EC.2 Acceptance thresholds for Provider-Dominance (PD) with linear revenue $R(L) = L$

Scenario	TR(O,P)	TR(R,P)	TR(O,NP)	TR(R,NP)
WL	10 (0%)	10 (0%)	10 (0%)	10 (0%)
WM	10 (0%)	10 (0%)	10 (0%)	10 (0%)
WH	10 (0%)	45 (39%)	10 (0%)	25 (15%)
EL	N/A	N/A	10 (0%)	10 (0%)
EM	N/A	N/A	10 (0%)	10 (0%)
EH	N/A	N/A	10 (0%)	10 (0%)

Table EC.3 Acceptance thresholds for Provider-Dominance (PD) with strictly convex revenue $R(L) = L^2/100$

Scenario	TR(O,P)	TR(R,P)	TR(O,NP)	TR(R,NP)
WL	15 (17%)	15 (17%)	15 (17%)	15 (17%)
WM	25 (33%)	25 (33%)	25 (33%)	25 (33%)
WH	60 (56%)	60 (56%)	60 (56%)	60 (56%)
EL	N/A	N/A	15 (17%)	20 (25%)
EM	N/A	N/A	15 (13%)	15 (13%)
EH	N/A	N/A	70 (67%)	30 (22%)

minimum possible treatment length indicate stricter policies. Percentages shown in parentheses place each threshold in the context of its feasible range: for CD this is TR/MAX, and for PD this is $(TR-MIN)/(MAX-MIN)$. In all cases, the Green state operates under FCFS and does not apply any threshold.

The notation TR(X, Y) identifies the threshold for occupancy state X (O = Orange, R = Red) and slot type Y (P = popular, NP = non-popular). In equal-preference scenarios (EL, EM, EH), all slots are equally popular, so the P columns are not applicable.

We observe that in the CD case (Table EC.1), thresholds are often lower for popular slots in weighted-preference scenarios, particularly in Orange and Red states, reflecting a tighter acceptance criterion for high-demand slots. Non-popular slots tend to have thresholds close to the maximum possible, resulting in behavior similar to FCFS. For PD with a linear revenue function (Table EC.2), many thresholds are at the minimum possible value, meaning that almost all requests are accepted regardless of occupancy, except in certain high-demand cases such as WH popular slots in the Red state. When the revenue function is strictly convex (Table EC.3), thresholds are generally higher, showing a stronger preference for longer treatments.

Appendix C: Deterministic Integer Programming Formulation

To benchmark the stochastic model and compute an upper bound on achievable performance, we formulate a deterministic version of the single-provider problem. This offline formulation assumes full knowledge of the entire sequence of requests (arrival times, program durations, and preference sets) over the finite planning horizon. The resulting optimization problem selects a feasible subset of requests and assigns them to time slots so as to maximize total reward. The optimal value of this formulation provides a valid upper bound on the performance of any online policy under uncertainty.

Sets and indices. Let $\mathcal{I}$ denote the set of client requests (indexed by i), $\mathcal{J}$ the set of within-period time slots (indexed by j), and $\mathcal{T} = \{1, \ldots, T\}$ the set of calendar periods (indexed by τ).

Parameters. Each request $i \in \mathcal{I}$ is characterized by:

- Program length $L_i \in \mathbb{Z}_+$,
- Arrival period $t_i \in \mathcal{T}$,
- Preferred slot set $J_i \subseteq \mathcal{J}$,
- Reward $r_i \geq 0$ if accepted.

Decision variable. For each request $i \in \mathcal{I}$ and slot $j \in J_i$, let $x_{ij} \in \{0, 1\}$ be a binary variable equal to 1 if request i is assigned to slot j, and 0 otherwise.

Model formulation.

$$\max \quad \sum_{i \in \mathcal{I}} \sum_{j \in J_i} r_i \, x_{ij} \tag{EC.2}$$

$$\text{s.t.} \quad \sum_{j \in J_i} x_{ij} \leq 1, \qquad \forall i \in \mathcal{I}, \tag{EC.3}$$

$$\sum_{i \in \mathcal{I}: \tau \in \{t_i+1, \ldots, t_i+L_i\}} x_{ij} \leq 1, \forall j \in \mathcal{J}, \ \forall \tau \in \mathcal{T}, \tag{EC.4}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i \in \mathcal{I}, \ \forall j \in J_i. \tag{EC.5}$$

The objective function (EC.2) maximizes total reward from accepted requests. Constraints (EC.3) ensure that each request is assigned to at most one slot. Constraints (EC.4) enforce slot capacity: no slot may serve more than one client in any period, accounting for the L_i consecutive sessions induced by recurring assignments. Constraints (EC.5) enforce integrality.

Appendix D: Small-Scale Optimal Policy Analysis

This appendix presents a small-scale computational analysis of the MDP model in settings where the optimal policy can be computed exactly. The purpose of this analysis is to provide additional insight into the structure of the optimal policy in a controlled environment, and to complement the analytical results derived in Section 4.

D.1. Experimental Setup

We consider simplified instances of the scheduling problem with three slots and a small discrete set of possible treatment durations. In each period, a single request arrives and specifies both a requested slot and a treatment length. Each request is associated with exactly one slot, selected according to a fixed probability distribution, so that the preference set is of size one.

The treatment duration is drawn from a discrete set of three values (5, 10, and 15 periods). The requested slot is drawn according to a fixed probability distribution over the three slots. We consider three representative demand patterns: a uniform distribution $(1/3, 1/3, 1/3)$, a mildly imbalanced distribution $(0.5, 0.3, 0.2)$, and a strongly imbalanced distribution $(0.8, 0.15, 0.05)$.

For each demand pattern, we compute the optimal policy under both reward structures introduced in Section 4: a client-dominance setting with constant reward, and a provider-dominance setting with $R(L) = L^2/100$, resulting in six instances in total.

For these small instances, the optimal policy is computed using exact dynamic programming. Due to the exponential growth of the state space, this approach is only computationally feasible for very limited problem sizes. In particular, extending the model even to four slots already leads to significant computational challenges, making exact solution impractical.

D.2. Results

The optimal policies obtained in these small-scale instances exhibit a clear and consistent structure. For each instance, the state space can be partitioned into regions in which requests are always accepted and regions in which they are always rejected. These regions are illustrated in Figure EC.1, where blue indicates acceptance and red indicates rejection.

A key observation is that the acceptance decision follows a threshold-type behavior with respect to the treatment duration L, in line with the analytical results derived in Section 4. In the client-dominance setting, longer requests are rejected, particularly in more popular slots, whereas in the provider-dominance setting shorter requests are rejected. These patterns are consistent across the different demand distributions considered.

In addition to illustrating the structure of the optimal policy, we also compared the optimal policy with a standard FCFS policy in the same small-scale settings. For each scenario, we generated 100 sample paths and evaluated the total reward obtained under the optimal policy and under FCFS. As shown in Table EC.4, the optimal policy consistently outperforms FCFS across all scenarios. While these improvements are observed in highly simplified settings, they further illustrate the value of anticipatory acceptance decisions even in small instances.

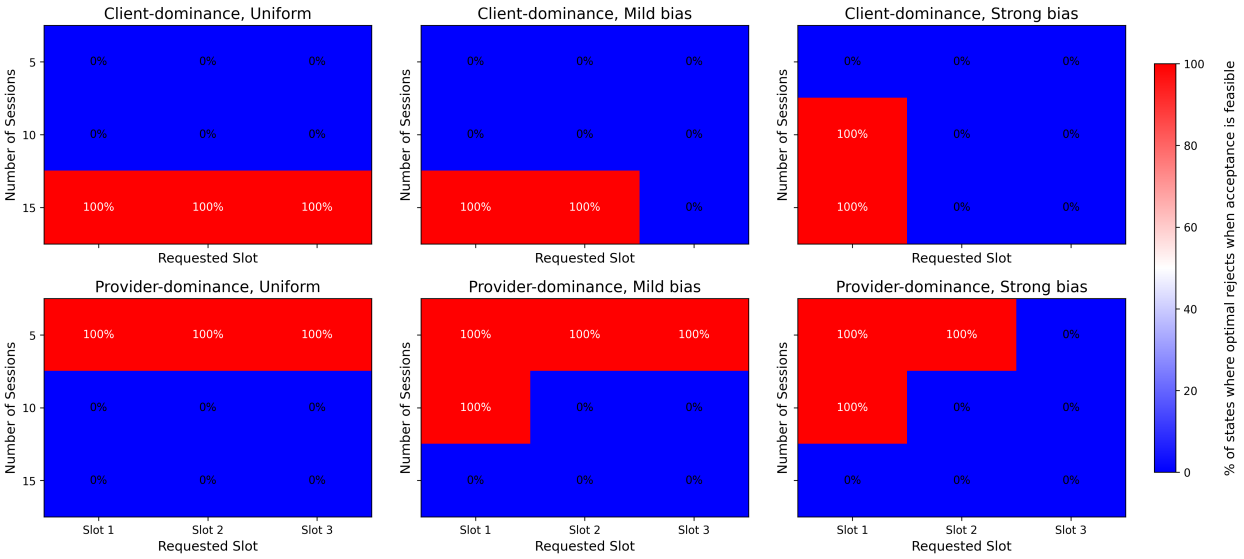

Figure EC.1 Optimal acceptance structure in small-scale instances. Blue regions indicate states in which requests are always accepted, while red regions indicate states in which requests are always rejected.

D.3. Discussion

The small-scale analysis presented in this appendix is intended to complement the analytical results in Section 4 by illustrating the structure of the optimal policy in settings where exact computation is feasible.

Table EC.4 Relative improvement of the optimal policy over FCFS in small-scale instances

Reward Structure	Slot Bias	Improvement over FCFS (%)
Client-dominance	Uniform	6%
Client-dominance	Mild bias $(0.5, 0.3, 0.2)$	6%
Client-dominance	Strong bias $(0.8, 0.15, 0.05)$	15%
Provider-dominance	Uniform	5%
Provider-dominance	Mild bias $(0.5, 0.3, 0.2)$	5%
Provider-dominance	Strong bias $(0.8, 0.15, 0.05)$	7%

The observed threshold-type behavior is fully consistent with the analytical characterization, supporting the interpretation of acceptance decisions as driven primarily by treatment duration and slot popularity.

At the same time, these experiments highlight the rapid growth of the state space, which limits the applicability of exact dynamic programming methods even in moderately sized instances. This observation further motivates the structure-based heuristic approach developed in Section 5, which aims to capture the key decision logic while remaining computationally tractable in larger settings.

References

Bertsekas D (2012) *Dynamic programming and optimal control: Volume I*, volume 4 (Athena scientific).