Fundraising for Education: Optimizing Crowdfunding Platforms

Mohammad Reza Aminian
The University of Chicago, Booth School of Business, Chicago, IL, maminian@chicagobooth.edu

Rad Niazadeh
The University of Chicago, Booth School of Business, Chicago, IL, rad.niazadeh@chicagobooth.edu

Click here for the latest version

Abstract.

We study the online operations of donation-based crowdfunding platforms, motivated by and calibrated to the education platform DonorsChoose. Projects arrive over time with funding goals and hard deadlines and are funded all-or-nothing, i.e., only if their donations reach the goal in time. Donors also arrive over time, choose among the projects they are shown, and give random amounts; a project may need several donations to reach its goal. The platform decides which projects to admit and which to display to each donor, so as to maximize the value-weighted rate of funded projects, subject to minimum success-rate floors for a protected set of project types.

A central obstacle is *donor dilution*: left uncoordinated, donor attention spreads over too many live projects, and many of them miss their deadlines. Our policy, which we call Curate-Share, counters dilution in two stages. A fluid linear program first thins the field, committing donor attention type by type and funding level by funding level; its value upper-bounds every policy, so it serves as the benchmark. Three modules then realize the plan online. Maintenance keeps each admitted project on the plan's trajectory, which fixes each type's load at its planned rate and, with it, the shape of the distribution of the number of active projects. Display guarantees every admitted project a floor matching rate u_j, which makes its deadline-miss probability exponentially small in u_j. Admission control admits only what can be served at the floor, capping each type's active count at $K_j^{\square}$, while Display splits the type's planned rate evenly among its active projects. This even split is what sharpens the guarantee: since the load is critical, it makes the active count uniform over the admissible range, so admission blocks with probability of order $1/\min_j K_j^{\square}$, where the cap $K_j^{\square}$ grows with the market.

We show that the policy is $1-\tilde{O}(1/\kappa)$-competitive against the fluid optimum, where κ is the factor by which the market scales. We also prove a nearly matching upper bound: on a single-type deterministic-deadline instance, every policy, online or offline, loses $\Omega(1/\kappa)$, so the rate is order-optimal up to polylogarithmic factors. When calibrated to the fairness-constrained optimum instead, the same policy is simultaneously value-competitive and approximately fair, at no additional online cost. In an extension, sponsor capital that buys back donor-rationed projects raises the benchmark by up to 32% on calibrated data, and the policy keeps its competitive share of the larger optimum. On the full DonorsChoose calibration, the policy comes within one percentage point of the fluid frontier and outperforms every greedy heuristic we test by two to twelve percentage points, depending on how the platform weights value. Floors that bring the highest-poverty classrooms to parity cost about 2.3% of value under a mission weighting.

Key words: crowdfunding platforms; online resource allocation; assortment optimization; loss networks; fairness; fluid approximation

1. Introduction

Public schools in the U.S. face serious funding problems, and these problems are often worse for the schools that need the most support. In the 2019–20 school year, 94.8% of public-school teachers spent their own money on classroom supplies without reimbursement, $470 on average among those who did (National Center for Education Statistics, 2026). Across the country, high-poverty districts receive on average 5% less state and

local revenue than low-poverty districts, about \$800 per student (Morgan, 2022), and an estimated 54% of public school districts need to update or replace multiple building systems in their schools (US Government Accountability Office, 2020). These gaps matter for students: on average, a policy that increases spending by \$1,000 per pupil for four years improves test scores by 0.0316 standard deviations and college-going by 2.8 percentage points (Jackson and Mackevicius, 2024). Education crowdfunding platforms try to alleviate these issues by letting teachers raise money for their classrooms directly from the public. However, these platforms face many challenges of their own and fall short of funding the projects they receive. For instance, DonorsChoose reports that only around 67% of its projects are fully funded (DonorsChoose, 2026b). This shortfall motivates our work: we study how such a platform can use the limited attention of its donors to fund more of these projects, and to fund them where they are needed most.

Crowdfunding is an increasingly more common fundraising mechanism that lets a person or an organization raise money online from many small contributors who wish to support their project. It is now widely used to finance startups, creative projects, and social causes, to name but a few. Grand View Research values the global crowdfunding market, measured by revenue, at USD 2.1 billion in 2024 and projects USD 5.5 billion by 2030, a compound annual growth rate of 17.6% from 2025 to 2030 (Grand View Research, 2025).

Crowdfunding mechanisms differ in what contributors receive in return for their funds. Backers on reward-based platforms such as Kickstarter and Indiegogo receive a product or a perk; investors on equity-based platforms such as Wefunder and StartEngine buy securities in young companies (Huang and Ivanov, 2025); lenders on debt-based platforms such as Prosper and Upstart are repaid their debt along with some interest; and donors on donation-based platforms such as GoFundMe and DonorsChoose receive no financial return. From a different dimension, these platforms can be partitioned into two groups based on how and when they determine success or failure of the projects they manage. Under keep-what-you-raise (or "flexible") funding, as on GoFundMe, an organizer keeps every donation whether or not the goal is met, and a fundraiser stays open until the organizer closes it (GoFundMe, 2026, 2025). Under all-or-nothing (or "fixed") funding, a campaign collects its pledges only if they reach its goal by a hard deadline: Kickstarter campaigns last at most 60 days (Kickstarter, 2026), and DonorsChoose projects stay live for up to four months (DonorsChoose, 2026a). Indiegogo offered both until October 2025, when it moved every campaign to all-or-nothing funding (Indiegogo, 2025).

This paper studies the donation-based, fixed funding platforms, although some of the insights and results can carry over to other types as well. In particular, we focus on one of these platforms, namely, DonorsChoose, which is a non-profit donation-based platform for U.S. public-school classrooms. Since 2000 it has raised \$2 billion from 6.8 million donors, funding 3.6 million projects for more than a million teachers in about 93,000 schools (DonorsChoose, 2026b). Most of their requests come from high-need classrooms: in 2013, 54% of all project requests came from schools that DonorsChoose classifies as highest-poverty (Section 6.1). Funded projects raise student achievement, most in low-income schools (Keppler et al., 2022), and keep teachers in their schools and in the profession (Keppler et al., 2025).

Three parties are involved in this operation: teachers, donors, and the platform that connects them. Teachers post projects over time, each requesting specific classroom items with a funding goal and a hard deadline, all listed on the project's page. Donors also arrive over time, see an assortment of live projects, and give to the project they choose, in amounts of their choosing. As stated before, typical[1] projects on this platform follow the all-or-nothing scheme: DonorsChoose buys the requested items only when a project reaches its full goal before its deadline, and otherwise the project fails and the donations go back to the donors[2]. We model the platform as an online two-sided matching market with two levers: which new requests to admit, and which live projects to show each arriving donor. Projects and donors of finitely many types arrive as independent streams. A project of each type has a goal, a deadline and a value to the platform. Each arriving donor chooses from the displayed assortment and gives a random amount whose distribution may depend on the project's type and on how far it has come (Section 2).

The goal of such a mission-driven platform as DonorsChoose could be throughput, total revenue raised, or potentially be something different. It may value projects unequally, e.g., a request from a high-poverty Title I school may be prioritized above an otherwise identical one, and it may require protected categories to be funded at some minimum rate so that an efficiency-only rule does not starve hard-to-fund classrooms. We capture these priorities through a *value* weight $w_j \geq 0$ for each project type and minimum *success-rate* targets β_j for a protected set of types.

In this view, we center the study of this paper around the following questions:

- *How should a donation-based crowdfunding platform admit and display projects, in real time and under uncertainty, to maximize the value-weighted rate of funded projects?*
- *How should the platform account for the all-or-nothing structure of funding under the pressure of hard deadlines?*
- *Can simple, interpretable policies achieve provably near-optimal performance while respecting equity guarantees for protected project categories?*

1.1. Challenges

Many projects fail for a reason the platform can influence, which we call *donor dilution*. When similar projects are live at once, the donors who would have completed one of them are spread across all of them, and each runs out of time short of its goal. For example, if six live projects each need two more donations and only six donations arrive before their deadline, spreading the donations evenly funds none of them, while concentrating them two by two funds three. Which projects the platform keeps active, and which of them it shows to each arriving donor, therefore decide which classrooms get funded.

[1] DonorsChoose also offers Classroom Essentials lists, which follow a different funding process.

[2] or sometimes to the teacher as credit if it's from the teacher or their "friends and family" backers (DonorsChoose, 2026c).

A failed project disappoints everyone involved. The donors' gifts are not spent on the classroom they wished to support, the teacher's request goes unsatisfied, and the students go without the materials they needed. Failures can also erode trust on both sides. On DonorsChoose, donors whose first project fails are less likely to donate again (Althoff and Leskovec, 2015). A teacher with a failed experience may never again post another project. Since funded projects raise student achievement and keep teachers in their schools (Keppler et al., 2022, 2025), a project that fails also forgoes these gains.

Coordinating donors is hard because of how a project gets funded. A project needs several donations to reach its goal, and each donation is random — its size may vary depending on the type and progress of the project. The project earns its value only if it reaches its goal before a hard deadline; a project that ends short — even by a single dollar — earns nothing. The platform can steer donors only through the assortments it shows them, and donors choose among what they see according to their own tastes, so the projects on the platform compete for the same pool of donor attention. The platform dynamically makes these choices in real time, without knowing which projects and donors will arrive in the future.

Existing models do not capture this combination of features. Many models of crowdfunding study a single campaign: Alaei et al. (2016) and Zhang et al. (2023), for example, study the stochastic progress of a single campaign assuming an exogenous stream of backers arrives over time and then provide guidelines on how fundraisers should design their campaign parameters; Du et al. (2022) also study the timing of a contingent stimulus for a given campaign whose backers pledge according to the campaign's expected success. There are also a few other works that consider platform-level control problems. Nevertheless, they all assume that a fixed set of projects is given to the platform in advance and then utilize the platform's control to affect the backers' behavior over time. Specifically, Hou et al. (2025) study the staff-pick problem, in which the platform can irrevocably assign a campaign, at some point during its lifetime, a badge that boosts its success. Song et al. (2022) come closest: they recommend DonorsChoose projects to returning donors, week by week, to maximize total funding under an estimated model of donor utility. Their heuristic solves a mixed-integer nonlinear program (MINLP) and treats the funding chance of a campaign, given its current state, as exogenous, i.e., unaffected by the platform's future recommendations. In contrast, our model captures the real-time stochastic arrival of projects and donors and accounts for the direct effect any policy may have on each project's funding trajectory. Furthermore, our policies enjoy performance guarantees against a fluid relaxation benchmark that are tight up to polylogarithmic factors. In Liang et al. (2026) the platform is given a fixed budget to allocate over time among a known set of projects, while each project receives its own independent stream of donors, making seed money the platform's only lever, so no project is admitted, displayed or discarded. As the seed money is the only element coupling different projects, they take the Lagrangian relaxation approach, an established method for handling weakly coupled dynamic programs. Using this relaxation, they construct a policy that respects the budget only in expectation, run it until the budget runs out, and show that its regret is sublinear in the number of projects. In our problem, by contrast, projects arrive stochastically and draw on a shared pool of arriving donors, whom the platform steers through its curation and display decisions. In Section EC.2, we also study

sponsor capital as another lever of the platform alongside the curation and display decisions, under a stationary long-run budget constraint, and show that with proper modifications, our results extend to this setting as well.

Beyond crowdfunding, the models closest to ours lie in the dynamic matching literature. Dynamic matching and queueing models with impatient agents (Aouad and Sarıtaç, 2022; Aveklouris et al., 2025; Liu et al., 2026) pair each arriving agent with at most one other agent, and an agent who waits too long leaves. In the matching queues of Gurvich and Ward (2014), a match may include at most one request from each type (a hypergraph matching), and once a match is made, all of its requests leave their queues at once; requests never abandon, and the goal is to minimize holding costs. In our model, by contrast, a project needs several partial, random donations, and it earns its value only if their sum reaches the goal before a deterministic deadline. Matching several donors to each project before its deadline, as both sides arrive over time, likewise differs from the one-to-one matching studied in the online matching, prophet inequality, and online contention resolution literature (Section EC.1).

1.2. Contributions and Results

Our contributions are in both the modeling aspect and the technical framework, which we elaborate on below.

The model. We introduce a model of donation-based crowdfunding that captures the key operational features of the application (Section 2). Projects and donors of finitely many types arrive over time. Each donor chooses from the assortment the platform shows and gives a random amount whose distribution depends on the project's type and funding level. A project succeeds only if its donations reach its goal by its deadline, in which case it earns a type-specific value for the platform (capturing throughput, revenue, or any other priority weighting); protected types must succeed at minimum rates. In an extension, sponsor capital can also advance projects and, like donations, is charged only on success. These features come from the application, and together they place the problem outside the standard models of online and dynamic matching, as discussed in Section 1.1.

The algorithmic framework and its guarantees. Our main technical contributions are as follows:

(i) A fluid benchmark that thins the market (Section 3). To measure performance, we consider the competitive ratio against a fluid linear program that we construct for this problem: a time-aggregate of the market, with no age variable, that thins the field by committing donor attention type by type and level by level. Its value upper-bounds every feasible policy, including clairvoyant policies that know all future arrival times and types in advance. We then use the solution of this fluid program to guide our online policy.

(ii) A near-optimal dynamic policy: a Triad Ensemble with balanced sharing, which we call Curate-Share (Section 4). Our online algorithm is composed of three interlocking modules: Maintenance, Display and Admission control. Together, they run the plan of the fluid LP against the live stream of projects and donors (Section 4.4 states the policy in one place). The policy is $1-\tilde{O}(1/\kappa)$-competitive against the fluid optimum as the market grows by a factor κ (Definition 2, Theorem 3), and this rate is tight up to polylogarithmic factors, since on a single-type deterministic-deadline instance every policy, online or offline, loses $\Omega(1/\kappa)$ (Proposition 2). Moreover, when calibrated to the fairness-constrained optimum, the same policy is *bi-criteria* (Corollary 2), i.e.,

simultaneously competitive for value and approximately fair, with any price of fairness borne by the fairness-constrained benchmark itself.

(iii) Sponsor capital (Section EC.2). Sponsors can add money of their own, for instance by matching donations (DonorsChoose, 2026c). We let the platform spend a capital budget to advance funding levels without donors. In our model capital is charged like donations, on success and never on failure, so the (opportunity) cost of finishing a project depends on how much capital was already spent on it, and its funding level alone no longer determines what it is worth to finish. Adding this spent capital to the state restores the missing information. On this enlarged state we again construct a modified fluid LP that upper-bounds every budget-feasible policy when the fairness floors are dropped, and every fair one when they are in place; the same Curate-Share policy of Section 4 (with proper adjustments to the maintenance module) attains it up to the same $1-\tilde{O}(1/\kappa)$ loss (Theorem EC.2).

1.3. Sketch of the techniques

Our framework first makes a plan and then executes it as closely as it can. The fluid program treats arrivals as steady flows and plans how to allocate donor attention by project type and funding level; its value upper-bounds every policy, and we use it as the benchmark. The program does not see time: a real project must collect its donations before its deadline, and the number of live projects as well as the amount of donations reaching them fluctuates with random arrivals on both sides. Our Curate-Share policy closes this gap with three modules, each supplying one property the others need. This design and its analysis are our main technical contribution: each admitted project gets a phase-type funding time with an exponentially small tail, while each type's share of the donor stream is split among however many projects of that type are active.

Maintenance runs each admitted project on a Markov chain over funding levels whose continuation probabilities come from the plan, so each type uses the plan's donor attention in expectation and succeeds at the plan's rate. It thus fixes each type's load, and with it the shape of the distribution of the number of active projects (the *count*), but on its own it says nothing about timing. Display supplies the timing: it dynamically chooses each donor's assortment so that every admitted project is matched at a guaranteed minimum rate, which makes the chance of missing the deadline exponentially small in that rate. Display can keep this promise only for a bounded number of active projects, and Admission control keeps the count within that bound: it caps each type's active count at $K_j^{\square}$, while Display, in the key step of the design, splits the type's aggregate planned rate evenly among its active projects. Because Maintenance makes the arriving workload match the planned rate exactly, admission becomes a critically loaded processor-sharing loss network. The critical load fixes the shape of the count's stationary distribution, and together with the caps, it determines this distribution exactly: by insensitivity, whatever the cap and whatever the distribution of donation sizes, the type-j count is equally likely to be any of $0, 1, \ldots, K_j^{\square}$, so an arriving project is blocked with probability $1/(K_j^{\square}+1)$ under the stationary distribution. Fixing each individual project's donor rate instead of sharing the aggregate rate among all the active ones (the Triad with dedicated, constant rates, which we call Curate-Dedicate), under the same cap, would block with probability of order $1/\sqrt{K_j^{\square}}$, quadratically worse (Remark 8).

The three properties combine into the guarantee. A type-j project is admitted with probability $1-\alpha_j^{\text{block}}$, and once admitted it is funded in time with probability at least $\rho_j-\delta_j$, where ρ_j is the plan's success rate and δ_j the deadline-miss probability. Since the policy starts in its stationary state, this holds whenever the project arrives (Theorem 2). Summing over types gives the competitive ratio $\min_j(1-\alpha_j^{\text{block}})(1-\delta_j/\rho_j)_+$ against the fluid optimum (Corollary 2). As the market grows by a factor κ, letting the minimum match rate grow like $\log\kappa$ makes both losses $\tilde{O}(1/\kappa)$ (Theorem 3).

1.4. Numerical and managerial insights

We calibrate every primitive of the model to the public DonorsChoose data for 2013 (KDD Cup, 2014) and run the policy as specified (Section 6). Four findings stand out.

Performance. On the calibrated instance, Curate-Share, the policy we recommend, reaches 99.5 to 99.7% of the fluid optimum under each of the three value weightings we study (throughput, revenue, and a mission weighting by students reached), with its caps set by a per-type search. It beats the best greedy rule we test by 2.5, 7.7 and 11.7 percentage points, respectively. When donor supply falls threefold, it loses only 0.2 points under throughput weights, compared with 7.5 points for the best greedy rule.

Fairness is cheap. Surprisingly, even when the LP's plan uses the mission weighting in its objective, the unconstrained value-maximizing plan abandons nine project types, five of them protected; with the fairness floors enforced, all five are served again. Moreover, raising the protected types to the funded rate the non-protected ones achieved in the 2013 data costs only about 2.3% of value, which demonstrates that equity can be achieved without much loss in efficiency. The cost of fairness comes mostly from the lowered fluid benchmark and only a small part from the online policy. Even this cost remains small for all but the largest floors, since the benchmark's value is concave in the floor and, on our data, falls steeply only near the largest feasible floor (Section 6.6).

Sponsor capital. Platforms should allocate their sponsor capital in a manner complementary to what donors support on their own. A budget can fund projects that are important and critical from the platform's perspective but, for some reason, less popular with donors overall. Even more important, the platform can strategically fund projects at stages that are less attractive to donors, where individual gifts are smallest, to bring them to success, and this is where most of the capital goes. The benefit of this extra budget has a concave shape, so a little extra budget can improve the objective significantly: under a third of the budget that funds every project already buys almost half of the gain, and the same capital lowers the price of fairness, to zero once every project is funded (Section 6.7).

Robustness, and what to estimate well. We also study what happens when the parameters of our instance change. When the platform still observes the new true parameters and re-solves its plan based on them, every perturbation we test has negligible effect on the performance ratio of our policy compared to the new optimum. When these changes are not observed, and our policy keeps a plan built on the biased estimates, the loss depends on what parameter is erroneous: a 20% error in the arrival mix across project types costs 8.5 points, a 20% shortfall in donor supply 4.5 points and making every gift one \$20 unit smaller (17.5% less on average)

4.1 points, whereas a 20% shorter deadline costs only 0.05. Lastly, our robustness checks point to the most important factors to estimate accurately: the arrival mix across project types, the donor supply and the donation sizes, which should also be kept current, while deadlines can be estimated roughly.

Policy message. All-or-nothing funding makes this a harsh environment, with a highly nonlinear objective that pays nothing for a project that is 99% funded and pays in full only once it is fully funded. To achieve good performance in such an environment, it is better for the platform to be more conservative and less ambitious in the first place. Our algorithm employs this insight through two forms of curation. The first is an individual-level curation that uses the fluid program as a guide to dynamically discard projects throughout their progress, including, but not limited to, when they first arrive. The second is a system-level curation, which restricts the count vector (the number of active projects of each type) to a safe region; when admitting a new arrival would put the system outside that region, the platform discards the arrival as well.

Roadmap. Section 2 presents the model, and Section 3 the fluid program that serves as our benchmark. Section 4 builds the online policy and derives its guarantee, Section 5 summarizes the extension to sponsor capital, and Section 6 calibrates the model to DonorsChoose data and draws the managerial lessons; Section 7 concludes. Further related work (Section EC.1), the sponsor-capital extension in full (Section EC.2), omitted proofs (Section EC.3) and additional numerical analyses with the calibration details (Section EC.4) are in the electronic companion.

2. Model

In this section we explain the model formally. There are three parties that play a central role in the crowdfunding ecosystem's business model. On one side, we have teachers that arrive over time and submit projects, each with a determined goal and a deadline by which it needs to raise all the money it requires to be successful, or otherwise it fails; on the other side we have donors who also arrive online and will provide funding to the project of their choice; and last but not least, we have the crowdfunding platform that bridges these two parties together — via gathering and curating the projects from teachers and then displaying an assortment of these projects that are live to each arriving donor. Figure 1 illustrates this dynamic and Table 1 collects the notation. In the following we explain the role of each of these three parties in more detail.

2.1. Primitives

We state the primitives in four categories — the arrival streams, the platform's values, the donor choice model, and the payment distributions.

Projects and donors. The platform hosts projects and donors of finitely many types. Projects of type $j \in \mathcal{J} = \{1, \dots, J\}$ arrive as a Poisson process of rate $\lambda_j > 0$; a type-j project carries a funding goal $G_j \in \mathbb{Z}_+$, a hard deadline $D_j > 0$, and a value $w_j \geq 0$ that the platform earns only if the goal is reached before the deadline. The value w_j is general and captures what the platform values, such as a welfare or priority weight (the impact of a funded type-j project); it need not equal the dollar goal, though it may. Donors of type $i \in \mathcal{I} = \{1, \dots, I\}$ also arrive as a Poisson process of rate $\mu_i > 0$. All arrival streams are mutually independent.

Welfare weights and protected types. The general value weights $w_j \geq 0$ enter the objective directly: the platform maximizes the total value it can generate through successfully funding projects, i.e., $\sum_j w_j \cdot$ (# successfully funded type-j projects). The pure-throughput model ($w_j \equiv 1$) and the revenue model ($w_j = G_j$) are two common special cases. Separately, a protected subset $\mathcal{P} \subseteq \mathcal{J}$ carries minimum success-rate targets $\beta_j > 0$: protected types must be funded at per-arrival rate at least β_j (made precise in Section 2.3 and imposed on the benchmark in Section 3.1). These two priority levers (objective weights and floor constraints) can be used independently or simultaneously.

Choice model. Each donor type i comes with a *choice function*

$$\varphi_i \colon 2^{\mathcal{J}} \to \Delta(\mathcal{J} \cup \{\varnothing\}), \qquad \varphi_i(\varnothing \mid R) = 1 - \sum_{j \in R} \varphi_i(j \mid R),$$

where $\varphi_i(j \mid R)$ is the probability that a type-i donor presented with the displayed assortment $R \subseteq \mathcal{J}$ selects project type $j \in R$. We use the convention $\varphi_i(j \mid R) = 0$ for $j \notin R$, so that the outside-option formula $\varphi_i(\varnothing \mid R) = 1 - \sum_{j \in R} \varphi_i(j \mid R)$ is unambiguous and aggregate sums can be written over $j \in \mathcal{J}$ without restricting to R. The induced per-donor-type *choice polytope* is $\widehat{F}_i := \operatorname{conv}\{(\varphi_i(j \mid R))_{j \in \mathcal{J}} : R \subseteq \mathcal{J}\} \subseteq [0,1]^J$, and the aggregate *Minkowski donor polytope* is $\widehat{F}_+ := \sum_{i \in \mathcal{I}} \mu_i \widehat{F}_i$. The polytope $\widehat{F}_+$ represents all possible ways we can distribute the combined attention of all donor types across different project types.

Payment distribution. First of all, for simplicity of exposition, we assume that the goals as well as payments (donation amounts) are positive and integer-valued[3]. With this in mind, when any donor donates to a type-j project whose current funding level is $g \in \{0, 1, \ldots, G_j - 1\}$, the donation amount is drawn from a probability mass function $q_{j,g}(\cdot) \in \Delta(\mathbb{Z}_+)$ that depends only on the project type and the funding level (j, g). In particular, for tractability of the analysis, we assume that the payment distribution is *shared across donor types and does not depend on the age of the project*[4]. We write $\bar{q}_{j,g}(k) := \sum_{p \geq k} q_{j,g}(p)$ for the upper-tail mass.

We present our key assumptions below.

In its most general form, the model allows the platform to include any positive number of live projects of a type in the displayed assortment, and once a donor selects a type, the donor may choose among the displayed projects of that type according to an arbitrary distribution. Our policies do not need this generality: they display a single project of each displayed type, picked uniformly at random among its active projects, so a donor who selects that type donates to it.

ASSUMPTION 1 **(Donor behavior and payments).** *(a)* Type-only choice. $\varphi_i(j \mid R)$ *depends only on which project types are present in the displayed assortment, not on the number or the state of the displayed projects of each type.*

[3] Our technical results and performance guarantees extend, with minor modifications, to payments that are continuous or can be 0; the running time of our policies depends linearly or quadratically (depending on the exact setup under study) on the discretization granularity. For example, with \$1 steps, our base model requires solving an LP with on the order of G_j variables for each project type, which is fast for a platform whose typical goals are a few hundred dollars.

[4] This is a modeling restriction rather than a loss-free reduction: donor heterogeneity in *selection* is carried by the type-dependent choice functions φ_i, while donation *sizes*, conditional on selecting a type-j project at level g, are assumed to share a common distribution across donor types.

(b) Downward-closed implementable marginals. *Each $\widehat{F}_i$ is downward closed in $\mathbb{R}_+^J$: if $y \in \widehat{F}_i$ and $0 \le y' \le y$ componentwise, then $y' \in \widehat{F}_i$.*

(c) Payment homogeneity. *For every $j \in \mathcal{J}$ and $g \in \{0, \dots, G_j - 1\}$ the payment distribution $q_{j,g}$ depends on (j, g) alone; in particular it does not depend on the type of the donating donor, nor on the history or age of the project.*

In other words, from a project's perspective, we capture donor-side heterogeneity in the donors' choice models rather than in their payments; from a donor's perspective, we capture heterogeneity across project types in the donor's choice model, and the dependence on a project's state (its level, given its type) in the payment distribution for that project type at that level.

REMARK 1 (A COMMON CHOICE MODEL SATISFYING 1(A) AND 1(B)). An example of a common choice model satisfying both 1(a) and 1(b) consists of two layers. Give each donor type i attraction weights $v_{ij} > 0$ and write $\tilde{m}_j \le m_j$ for the number of type-j projects included in the assortment, out of the m_j active ones, and $\mathbb{I}_j := \mathbf{1}\{\tilde{m}_j \ge 1\}$ for whether any type-j project is available in it. A type-i donor shown assortment R picks a *type* by multinomial logit gated on availability,

$$\varphi_i(j \mid R) \;=\; \frac{v_{ij}\,\mathbb{I}_j}{1 + \sum_{k \in \mathcal{J}} v_{ik}\,\mathbb{I}_k},$$

and, having picked type j, picks one of the $\tilde{m}_j$ displayed projects of that type uniformly at random.

To see that 1(a) holds, note that the type-level choice probability depends on R only through which types it contains and not through how many projects of each type are shown; the second layer also expresses no preference among the displayed type-j projects, so the choice does not depend on the state of the projects of the chosen type either. Downward closure, Assumption 1(b), follows from the weak substitutability of the MNL choice model, which we discuss next. The last part, 1(c), is a restriction on payments rather than on choice, so the choice model neither delivers nor obstructs it, and any payment distribution $q_{j,g}$ shared across donor types completes the example.

We assume throughout that the choice model is *weakly substitutable*: for every donor type i, every $R \subseteq R' \subseteq \mathcal{J}$ and every $j \in R$, $\varphi_i(j \mid R) \ge \varphi_i(j \mid R')$, so enlarging the assortment never raises any type's selection probability. The model above satisfies this property, as a two-line argument shows. Fix $R \subseteq R'$ and $j \in R$: the numerator $v_{ij}\mathbb{I}_j$ is the same under both assortments, while $1 + \sum_{k \in R} v_{ik}\mathbb{I}_k \le 1 + \sum_{k \in R'} v_{ik}\mathbb{I}_k$ because every omitted term is non-negative, so the ratio is weakly larger under R. Likewise, when a type empties, $\mathbb{I}_k$ falls to zero and its term leaves the denominator, so every other type's probability weakly rises. When a type loses some but not all of its projects, $\mathbb{I}_j$, and hence its type-level probability, are unchanged, while each surviving project's share $1/\tilde{m}_j$ rises.

LEMMA 1 **(Weak substitutability implies downward closure; adapted from Feng et al. (2024)).** *A weakly substitutable choice model satisfies downward closure, Assumption 1(b).*

REMARK 2 (A SECOND ROUTE TO DOWNWARD CLOSURE). Assumption 1(b) can also be delivered by the platform rather than assumed of the donors. A platform that displays the full assortment and then discards a fraction of the selections given the realized choices can implement any coordinatewise smaller rate vector, so the implementable marginals are downward closed whatever the choice functions look like. This route needs no restriction on φ_i, but rather assumes a donor will leave (choose the outside option) if their selection is denied.

The reason behind the assumption 1(c) is structural. If payments depend on who gives, a project's completion trajectory becomes heterogeneous: at different funding levels it needs different donor types to fund it. Meeting a deadline would then require a guaranteed rate of donor attention from donor types that change dynamically with the project's trajectory, and delivering such guarantees requires admission control per donor type in place of the single central control on $\widehat{F}_+$ that Section 4.3 runs. Since a project would have to be admitted by every donor type whose attention its trajectory may need, delivering the deadline promise would sharply reduce the admission probability. We therefore deliberately capture donor heterogeneity through the choice model φ_i, where it does no such damage, rather than through the payment distribution. The assumption is also more plausible than it first looks when the platform observes donors only coarsely: the finer the platform's view of a donor, the more it could in principle condition on, and a model that sets the payment distribution at the (project-type, level) pair is a statement about what such a platform can observe and act on rather than about what potential variations can exist but are hidden from the platform.

2.2. Dynamics

The system evolves in continuous time and changes at three types of events: a project arrives and is admitted or rejected; a donor arrives, is shown an assortment, and may choose a project to donate to; and a project leaves by reaching its goal or by running out of time.

Horizon and initial state. The *primary model* of this paper has an infinite horizon: project and donor arrivals occur on $[0, \infty)$, and each admitted project continues service until its own deadline D_j (measured from its admission time). The initial project inventory is empty: $S(0) = \emptyset$. The objective $J(\pi)$ is then the $\limsup_{T\to\infty}$ of the time-averaged expected value (eq. (1)).

This formulation has a key structural advantage: levels change only at match epochs, so reaching a level and being matched into it are the same event with the same timestamp, and the flow counts up to time T satisfy the reach recursion (3) as an *exact equality* for any finite T and any policy — a Wald identity, since whether the kth match occurs by T is determined before that match's payment is revealed — with no initial-inventory or terminal-truncation corrections. This makes the upper-bound proof (Proposition 1) direct and elementary. The policy side is exact for the complementary reason: the algorithm is started in its stationary distribution (Section 4.3), so a project arriving at any point in time faces the identical admission probability and the per-arrival guarantee of Section 4.5 holds whenever it arrives.

At time $t \geq 0$ the live project pool is the multiset $S(t)$ of currently live projects, each identified by its type j, current level $g \in [0, G_j)$, and arrival time $a \in (t - D_j, t]$.

A *donor event* of type i at time t: the platform observes $S(t)$ and selects a displayed assortment $R \subseteq \mathcal{J}$ (by Assumption 1(a), the assortment can equivalently be viewed as a set of types rather than of individual projects). The donor selects $j \in R \cup \{\varnothing\}$ with probability $\varphi_i(j \mid R)$. Selecting type j means that the donor selects one of the displayed type-j projects to donate to[5]. A positive integer payment $p \sim q_{j,g}(\cdot)$ is drawn at the chosen project's current level g and the level updates to $\min(g+p, G_j)$[6].

A *project event* of type j at time t: the platform observes $S(t^-)$ and decides whether to admit the project (it then enters $S(t)$ with level 0 and deadline clock D_j) or to reject it.

Exit events. A project leaves in one of two ways: (success) its level reaches G_j, earning the platform value w_j; or (failure) its time to deadline reaches 0 with level $< G_j$, in which case it earns nothing and the donations are returned to donors — the widespread *all-or-nothing* practice of many crowdfunding platforms, including DonorsChoose, Kickstarter, and Indiegogo.

2.3. Policy, objective, and constraints

In the following we describe the general class of platform policies against which all our guarantees are stated.

General non-anticipating policy class. A non-anticipating policy π specifies, on each project arrival of type j at time t, an admission decision $\pi_{\mathrm{adm}}(t, S(t^-), j) \in \{0,1\}$, and on each donor arrival of type i at time t, a displayed assortment $\pi_{\mathrm{disp}}(t, S(t^-), i) \subseteq \mathcal{J}$ (together with which live projects of each displayed type to include, any positive number of them, possibly at random). Both rules are adapted to the natural filtration $\mathcal{F}_t$ generated, up to time t^-, by the project and donor arrival processes, the realized assortments displayed, the realized donor choices, and the realized payment amounts. Policies have no access to future arrivals, choices, or payment realizations. The set of all such policies is denoted Π_{na}. We also consider the *optimal clairvoyant online policy*, which knows the entire sequence of project and donor arrival times and types in advance but still observes neither the donors' choice realizations nor the payment realizations (each is drawn only when an assortment is displayed and a donation is made).

Objective. The objective is the long-run *value* rate

$$J(\pi) := \limsup_{T\to\infty} \frac{1}{T}\,\mathbb{E}_\pi\Big[\sum_j w_j N_j^{s,\pi}(T)\Big], \tag{1}$$

where $N_j^{s,\pi}(T)$ denotes the number of type-j projects successfully funded by time T. Let us also define the type-j success rate under π — the expected number of type-j projects funded per unit time up to T —

$$S_j^{\pi,T} := \tfrac{1}{T}\,\mathbb{E}_\pi\big[N_j^{s,\pi}(T)\big]. \tag{2}$$

The objective $J(\pi)$ is the long-run rate at which the platform accrues value w_j from funded type-j projects (the pure-throughput rate of funded projects when $w_j \equiv 1$, and the revenue rate when $w_j = G_j$). We take the long-run rate because at any finite horizon T the projects arriving shortly before T do not yet have their full deadline D_j of donor arrivals ahead of them; in the limit, this edge effect vanishes.

[5] The platform can choose which individual project, or subset of projects, of type j to include in the assortment; in our policies, it displays a single type-j project, chosen uniformly at random among the active ones.

[6] The actual payment is capped at $\min(p, G_j - g)$ so that the level does not exceed the goal.

Fairness targets. Recall the protected subset $\mathcal{P} \subseteq \mathcal{J}$ with minimum success-rate targets $\beta_j > 0$, $j \in \mathcal{P}$ (Section 2.1). We call a policy π *fair* if

$$\liminf_{T\to\infty} S_j^{\pi,T}/\lambda_j \;\geq\; \beta_j \qquad \text{for every } j \in \mathcal{P}.$$

The platform seeks a fair policy of high value: it maximizes (1) subject to these fairness floors. The pure-value (no-fairness) model is the special case $\mathcal{P} = \varnothing$, and the classical pure-revenue model additionally sets $w_j = G_j$. The next section builds the benchmark: a fluid program that upper-bounds the long-run average value of every non-anticipating policy, and even of the clairvoyant one, and whose optimal solution feeds into our proposed policy.

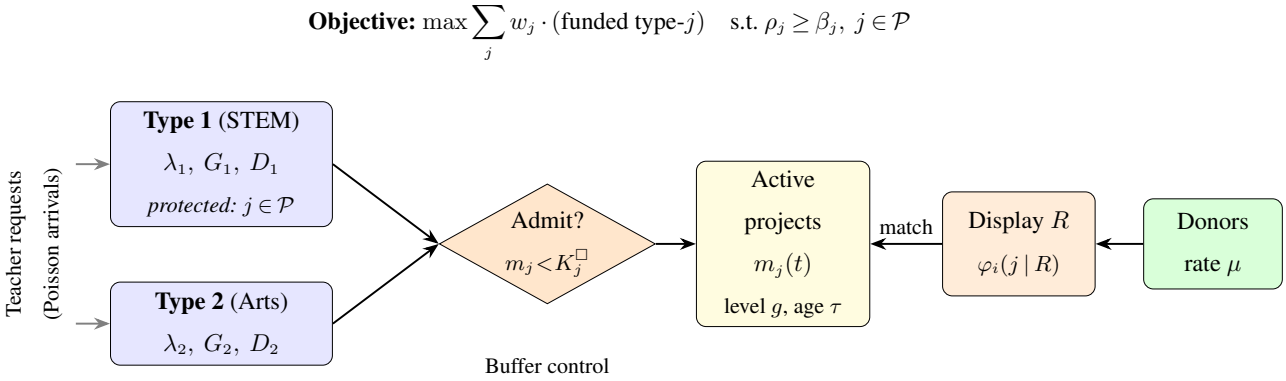


Figure 1 The platform as a matching system. Type-j projects (teacher requests) arrive as independent Poisson streams of rate λ_j, each with funding goal G_j and deadline D_j; an arrival is admitted only if the type-j count stays within the commitment box, $m_j < K_j^{\square}$. Admitted projects join the active pool, each tracked by its level g and age τ, giving the count vector $m_j(t)$. On each donor arrival (aggregate rate μ) the platform shows an assortment R, and a type-i donor selects type j with probability $\varphi_i(j \mid R)$, routing a match to a uniformly chosen active type-j project. The platform maximizes the value-weighted funded-project rate $\sum_j w_j \cdot$ (funded type-j) subject to fairness floors $\rho_j \geq \beta_j$ for protected types $j \in \mathcal{P}$.

Symbol	Meaning	Defined in
$\lambda_j,\ G_j,\ D_j,\ w_j$	type-j arrival rate, funding goal, deadline, value weight	Section 2.1
$\mu_i,\ \varphi_i(\cdot \mid R)$	type-i donor rate and choice function	Section 2.1
$\widehat{F}_+ = \sum_i \mu_i \widehat{F}_i$	aggregate donor polytope (implementable match-rate vectors)	Section 2.1
$q_{j,g},\ \bar{q}_{j,g}$	payment distribution at level g and its upper tail	Section 2.1
$\mathcal{P},\ \beta_j$	protected types and success-rate floors	Section 2.3
$M_j(g),\ \Pi_j(g)$	fluid match rate and reach rate at level g	Section 3
$c_j(g),\ a_j(g),\ b_j(g)$	continuation probability; active and entry probabilities	(9)
ρ_j	per-arrival success rate; $\mathrm{OPT}_{\mathrm{LP}} = \sum_j \lambda_j w_j \rho_j$	(12)
$\bar{K}_j^{\mathrm{don}},\ \bar{y}_j = \lambda_j \bar{K}_j^{\mathrm{don}}$	mean donations per type-j arrival; reserved attention rate	(15)
$u_j,\ K_j^{\square} = \lfloor \bar{y}_j/u_j \rfloor$	floor matching rate; box cap of type j	Section 4.2; (23)
$\mathcal{K}^{\square} = \{n \in \mathbb{Z}_+^J :\ n_j \leq K_j^{\square} \text{ for every } j\}$	the commitment box as a set of counts	Section 4.3, before (23)
$\alpha_j^{\mathrm{block}},\ \delta_j$	blocking probability; deadline-miss probability	Section 4.5; Section 4.2
γ	floor level of the fairness sweep, $\beta_j \equiv \gamma \cdot \bar{\rho}_{\mathrm{unprot}}^{\mathrm{emp}}$	Section 6.6

Table 1 Summary of notation.

3. The fluid filter

The first move against dilution is made before the system runs. Rather than let every arriving project compete for attention, the platform thins the field: it commits, type by type and level by level, to how much donor attention each kind of project will receive, and admits projects only up to what that commitment can serve. The instrument is a fluid relaxation of the market — a small linear program whose solution, which we refer to as the thinning plan, guides our policy. Its match rates $M_j(g)$ say how much attention survives the filter at each funding level; the ratios $c_j(g)$, the *continuation probabilities*, say what fraction of projects reaching a level stay in the plan; and the per-arrival success rate ρ_j is the plan's promise for each type. The program asks only that the plan be feasible for the donors in a fluid sense, and in return it upper-bounds the value of *every* non-anticipating policy, and even of the clairvoyant policy, so a ratio earned against it is a ratio against the true optimum. It relaxes the true problem in exactly two ways: it credits a funding transition without asking whether the deadline had passed, and it imposes donor feasibility on time averages rather than at every instant. These two relaxations are the main sources of its optimism, and Section 4 accounts for them separately — the deadline through a guaranteed minimum matching rate, the instantaneous feasibility through admission control. A key factor enabling such a condensed LP is the Markovian structure of each project's progress: the age and history at which a match happens wash out, leaving a finite recursion with no time variable.

3.1. The fluid relaxation

The fluid relaxation tracks, for each type j and transient level $g \in \{0, \dots, G_j - 1\}$, two long-run rates: the rate $M_j(g)$ at which type-j projects are *matched* (receive a donation) while at funding level g, and the rate $\Pi_j(g)$ at which they *reach* level g. A match at level r carries the project to level $r + p$ with payment probability $q_{j,r}(p)$, so these two families of rates are tied by a forward recursion, while time, including the deadline, is abstracted away from the program.

Variables. The decision variables are the match rates $M_j(g) \geq 0$ for each $j \in \mathcal{J}$ and $g \in \{0, \dots, G_j - 1\}$. The *reach rate* $\Pi_j(g)$ is the derived quantity

$$\Pi_j(0) := \lambda_j, \qquad \Pi_j(g) := \sum_{r=0}^{g-1} q_{j,r}(g-r)\, M_j(r) \quad (1 \leq g \leq G_j - 1). \tag{3}$$

Every arrival reaches level 0, giving $\Pi_j(0) = \lambda_j$; a project reaches a higher level g precisely by being matched at some lower level r and drawing the payment $g - r$ that lands it there, which is the sum in (3).

The program. The objective is the value rate of new successes,

$$\max \sum_{j \in \mathcal{J}} w_j\, S_j, \qquad S_j := \sum_{g=0}^{G_j - 1} \bar{q}_{j,g}(G_j - g)\, M_j(g), \tag{4}$$

subject to

$$\text{(Project Availability)} \quad M_j(g) \leq \Pi_j(g), \qquad j \in \mathcal{J},\ 0 \leq g \leq G_j - 1, \tag{5}$$

$$\text{(Donor Availability)} \quad \left(\sum_{g=0}^{G_j - 1} M_j(g)\right)_{j \in \mathcal{J}} \in \widehat{F}_+ = \textstyle\sum_{i \in \mathcal{I}} \mu_i \widehat{F}_i, \tag{6}$$

$$\text{(Protection)} \quad \rho_j := \frac{S_j}{\lambda_j} \geq \beta_j, \qquad j \in \mathcal{P}, \tag{7}$$

$$\text{(Non-neg)} \quad M_j(g) \geq 0, \qquad j \in \mathcal{J},\ 0 \leq g \leq G_j - 1, \tag{8}$$

with Π_j given by (3). The objective reads off the value rate: a project matched at level g succeeds when its "nominal" payment overshoots the goal (i.e., the actual payment brings it to the goal), which happens with probability $\bar{q}_{j,g}(G_j - g)$, and it then contributes w_j. Constraint (Project Availability) says a type-j project cannot be matched at level g more often than it reaches g; its slack $\Pi_j(g) - M_j(g) \geq 0$ has a physical interpretation as the rate of type-j projects that reach level g and leave it unmatched, whether by deadline expiry, by admission rationing, or by the deliberate thinning of the plan; the ratio $c_j(g) := M_j(g)/\Pi_j(g) \in [0,1]$ is the continuation probability developed in Section 3.2. Constraint (Donor Availability) is donor-side feasibility — the vector of total per-type match rates lies in the aggregate donor polytope $\widehat{F}_+$. Lastly, (Protection) imposes the protected success floors.[7]

The two relaxations. The four constraints displayed above define the *fairness-constrained* program (P_F), with optimum OPT_F; deleting (Protection) gives the *unconstrained value* program (P), with optimum $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}} \geq \mathrm{OPT}_F$, the two coinciding when $\mathcal{P} = \varnothing$. The algorithm of Section 4 is calibrated to an optimal solution of the program that carries the floors, so the value and fairness guarantees are delivered by the *same* policy (Section 4.5). Throughout, $\mathrm{OPT}_{\mathrm{LP}}$ denotes the optimum of the *driving* program, i.e., the relaxation to which the algorithm is calibrated and whose optimal solution fixes the rates ρ_j of (7): it is OPT_F when the floors are imposed and $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$ when $\mathcal{P} = \varnothing$. Every competitive ratio below is stated against $\mathrm{OPT}_{\mathrm{LP}}$. Proposition 1 bounds every policy by $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$ and every fair one by OPT_F, and the development through Section 4.5 applies verbatim to either calibration.

REMARK 3 (FEASIBILITY OF THE FLOORS). Every fairness statement below presumes that (P_F) is feasible, meaning that the protected targets $(\beta_j)_{j\in\mathcal{P}}$ are jointly achievable by some fluid allocation, so the required protected match flow fits within the donor polytope $\widehat{F}_+$.

The following proposition formally shows that the program is indeed an upper bound on any non-anticipating policy, as well as the optimal clairvoyant online policy.

PROPOSITION 1 **(The program is an upper bound).** *For every policy $\pi \in \Pi_{\mathrm{na}}$ — and, more strongly, every clairvoyant policy that knows all future arrival times and types but observes each donor choice and payment only when it is realized — $J(\pi) \leq \mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$. Moreover, $J(\pi) \leq \mathrm{OPT}_F$ for every non-anticipating fair policy. The latter statement presumes that (P_F) is feasible, as set out in Remark 3; otherwise, OPT_F does not exist.*

[7] The program can be solved in time polynomial in its size whenever assortment optimization can, for each donor type's choice model: a linear objective over $\widehat{F}_+$ separates into one assortment optimization per donor type, so the ellipsoid method, with these optimizations as its oracle, solves the program (Grötschel et al., 1988). Under MNL, the sales-based formulation of Gallego et al. (2015) writes (Donor Availability) with polynomially many linear constraints, which is how we solve it in Section 6.

Proof sketch. Fix a policy and a horizon T, and let $M_j^{\pi,T}(g)$ be its realized match rate at level g. (Project Availability): positive payments mean that a project reaches each transient level at most once, a payment is drawn only at the match epoch that produces it, and a project must reach a level before it is matched there. (Donor Availability): every match is preceded by one donor selection, and by Assumption 1(a) the choice depends on the assortment only through the types in it, so each donor type's realized selection vector is a convex combination of its choice vertices — for *any* policy, clairvoyant included, because none of them foresees a realized choice — and, since matches cannot exceed selections, downward closure (Assumption 1(b)) carries them into $\widehat{F}_+$. Objective: a success is a match whose (final nominal) payment overshoots the goal (i.e., the actual level reaches the goal), so the realized value rate is the program's objective evaluated at the realized match rates. Section EC.3 gives the argument in full.

REMARK 4 (SCOPE OF THE BENCHMARK). The proof accounts for donor flow only at the type level — matches cannot exceed selections — and never references how a donation is routed *within* a type. The bound $J(\pi) \le \mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$ therefore holds for policies with arbitrary within-type routing, including age- or level-dependent routing, not only for the uniform routing convention we use for our policies; this is what enables the relaxation to also bound the heuristics we introduce in Section 6.

REMARK 5 (THE SERVED SET). Write $\mathcal{J}^\rho := \{j : \rho_j > 0\}$ for the types the fixed optimum funds at a positive rate, and drop the rest. Dropping them is without loss, i.e., we obtain the same LP solution and value on the reduced instance. A type with $\rho_j = 0$ contributes a zero term to $\mathrm{OPT}_{\mathrm{LP}} = \sum_j \lambda_j w_j \rho_j$, and its protection floor, if any, must also be zero. Also note that an LP solution matching such a type at low levels and abandoning it before the goal is strictly improved by releasing those matches to a type with $\rho > 0$ whenever donors are scarce, so $\rho_j = 0$ forces $M_j \equiv 0$ and hence $c_j(0) = 0$, and the solution essentially ignores that type. All statements below are therefore made over $\mathcal{J}^\rho$, and every benchmark is $\sum_{j \in \mathcal{J}^\rho} \lambda_j w_j \rho_j$, which is $\mathrm{OPT}_{\mathrm{LP}}$. Note that the fairness-constrained program never drops a protected type: (Protection) forces $S_j \ge \lambda_j \beta_j$ with $\beta_j > 0$, hence $\rho_j \ge \beta_j > 0$ for every $j \in \mathcal{P}$, so under (P_F) the dropped set is entirely unprotected by construction.

3.2. Structural quantities

Fix an optimal solution $(M_j(g))$ of the driving program, with the reach rates $\Pi_j(g)$ it induces through (3). Three structural quantities read off this optimum drive the algorithm.

(S1) Continuation probabilities $c_j(g)$: where the plan thins. Define

$$c_j(g) := \frac{M_j(g)}{\Pi_j(g)} \in [0,1], \qquad g \in \{0, \dots, G_j - 1\}, \tag{9}$$

with the convention $c_j(g) := 0$ if $\Pi_j(g) = 0$ (immaterial, since no project ever reaches g). The ratio lies in $[0,1]$ because of constraint (5). The complement $1 - c_j(g)$ can be thought of as abandonment at entry to g, consuming no donor there.

(S2) Visit probabilities $b_j(g)$, $a_j(g)$: the donor budget per arrival. Define

$$b_j(g) := \frac{\Pi_j(g)}{\lambda_j}, \qquad a_j(g) := c_j(g)\, b_j(g) = \frac{M_j(g)}{\lambda_j}, \qquad g \in \{0, \dots, G_j - 1\}. \tag{10}$$

Here $b_j(g)$ is the *tentative-entry probability*—the probability that the chain arrives at g before the continuation coin, with success probability $c_j(g)$, is flipped—and $a_j(g)$ is the *active-state visit probability*—entry and survival combined. Donor consumption at g is proportional to $a_j(g)$, not $b_j(g)$. Since $\Pi_j(0) = \lambda_j$, we have $b_j(0) = 1$ and $a_j(0) = c_j(0)$; for $g \geq 1$, the throughput identity yields the recursion

$$b_j(g) = \sum_{r=0}^{g-1} a_j(r)\, q_{j,r}(g - r), \qquad a_j(g) = c_j(g)\, b_j(g), \qquad g \in \{1, \dots, G_j - 1\}. \tag{11}$$

(S3) Success rate per arrival ρ_j: what the plan promises a type. By the objective (4),

$$\rho_j := \frac{S_j^*}{\lambda_j} = \sum_{r=0}^{G_j - 1} a_j(r)\, \bar{q}_{j,r}(G_j - r) \in [0, 1], \tag{12}$$

where the second equality uses $M_j(g) = \lambda_j a_j(g)$. Success at G_j requires reaching (or overshooting) G_j, so the success rate uses the tail mass $\bar{q}_{j,r}(G_j - r)$ rather than the exact-hit kernel. Summing over types, $\mathrm{OPT}_{\mathrm{LP}} = \sum_j \lambda_j w_j \rho_j$.

EXAMPLE 1 (THINNING IN A TWO-LEVEL MARKET). To see the plan's logic concretely, consider a single project type with goal $G = 2$, deadline D, unit payments ($q_g(1) = 1$ for $g \in \{0, 1\}$, so every project needs exactly two donations), arrival rate $\lambda = 10$ per day, and a single donor type that always selects the displayed project when one is shown, with donor arrival rate $\mu = 12$ per day. Each funded project consumes two donations, but only 12 donations arrive per day against 10 project arrivals demanding 20. Left uncoordinated, all ten daily projects collect donations at an average of 1.2 per project over its lifetime, so most reach level 1 and stall, funding almost nothing despite full donor utilization. The fluid program instead solves $\max \bar{q}_0(2)M(0) + \bar{q}_1(1)M(1)$ subject to $M(0) \leq 10$, $M(1) \leq \Pi(1) = M(0)$, and $M(0) + M(1) \leq 12$; with unit payments $\bar{q}_0(2) = 0$ and $\bar{q}_1(1) = 1$, so the objective is the completion flow $M(1)$, and the optimum sets $M(0) = M(1) = 6$. The plan reads: continue $c(0) = M(0)/\lambda = 0.6$ of arriving projects at level 0, and continue *all* of those into level 1 ($c(1) = M(1)/\Pi(1) = 1$). The thinning is entirely at the gate: six projects per day are served to completion ($\rho = 0.6$) and four are let go immediately, rather than ten being half-served.

Ideally, we would design a policy that follows the plan of the LP exactly. However, the fluid LP suppresses time and any stochasticity in the arrivals. In reality, the challenge arises as projects arrive at random times, and if the LP's plan directs us to fund a project, we need to make sure that it receives enough donations before its deadline; otherwise, it is worth nothing. The next section builds an online policy that addresses this challenge and implements the fluid plan in a stochastic, dynamic environment.

4. The online algorithm: a Triad Ensemble

The fluid filter is an offline plan: it says, for each type and funding level, what fraction of the projects should be matched. Realizing it against a live, stochastic stream of donors and projects, under the pressure of the deadline, is the online part of the job, and our policy handles it in three modules, which we call the Triad, that must all work together at the same time. Each secures one property another module needs and, alone, fails in a way another of the three modules repairs — so the core idea lies in what these modules are and how they close each other's gaps. Maintenance (Section 4.1) steers each admitted project along the fluid plan's trajectory, so each project type consumes exactly the plan's expected number of donations and is successfully funded at exactly the plan's rate; what it does not control is timing. Display (Section 4.2) supplies the clock: it guarantees every admitted project a floor match rate u_j from admission, making its funding time a phase-type random variable whose tail decays exponentially in u_j, thereby addressing the donor-side stochasticity of the problem. The caveat is that, because the supply of donors is limited, Display cannot guarantee this floor rate to an arbitrarily large number of active projects. In particular, the Maintenance module only determines the load of the system, which it keeps critical, just at the boundary up to which Display can deliver its promise. This fixes the shape of the distribution of the count (the number of active projects of each type); together with the box within which the count must stay, this shape determines the distribution exactly. Without such a box, the stochastic nature of project arrivals would make the count fluctuate, causing problems for Display whenever the count falls outside the boundary. Admission control (Section 4.3) is in charge of converting the ex-ante (average) promise into an ex-post one, and does so with a blocking probability of order $1/K_j^{\square}$. None of the three modules can function alone: Maintenance needs Display's timing but in return provides the ex-ante promise on the system load and project count, Display needs Admission control's ex-post promise on the count but provides the deadline guarantee, and Admission control needs the ex-ante promise of Maintenance to achieve the ex-post promise with only a small blocking loss. Each module states its guarantees where they are used; detailed proofs are collected in Appendix EC.3. Figure 2 shows the dependence; Section 4.4 and Figure 3 state the policy in one place once its modules have been introduced.

4.1. Maintenance

Maintenance is the module that keeps admitted projects on the plan. It encodes the filter's continuation probabilities into a per-project Markov chain — including a null state for the projects the plan drops — so that a project run by this chain consumes, in expectation, exactly the donor budget the plan allotted it and is funded at exactly the plan's rate (Lemma 3). Its gap is timing: the chain says nothing about whether the donations it needs arrive before the project's deadline.

The Survival Chain. For each project type j, define a discrete-time Markov chain on the state space $\mathcal{S}_j = \{0, 1, \ldots, G_j - 1, G_j, \star\}$, with transient states $\{0, \ldots, G_j - 1\}$ and absorbing states G_j (success) and $\star$ (null).

Upon admission, a type-j project is initialized in active state 0 with probability $c_j(0)$, and in absorbing state $\star$ with probability $1 - c_j(0)$. This coin is part of the *initial distribution* of the per-project process—flipped at admission with no donor consumed—rather than a transition of the chain.

Whenever a donor match happens at some active state g, the following steps are taken:

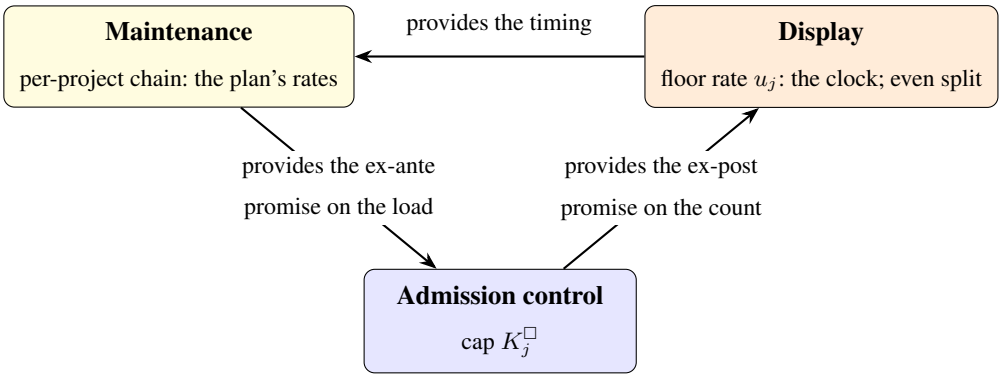


Figure 2 The three modules depend on one another. Maintenance needs Display's timing and in return supplies the ex-ante promise on the system load and project count; Display needs Admission control's ex-post promise on the count and in return supplies the deadline guarantee; Admission control needs Maintenance's ex-ante promise to deliver the ex-post one with only a small blocking loss. None of the three can function alone.

- Draw a positive payment $p \sim q_{j,g}(\cdot)$.
- Compute the tentative destination $h := \min(g+p, G_j)$.
- If $h = G_j$, the project is absorbed at G_j (success).
- If $h < G_j$, the project survives into active state h with probability $c_j(h)$; otherwise, it transitions immediately to $\star$ (without consuming a further donor at h).

Note that the continuation coin is attached to the destination level h, not the origin g, and that no coin is used at G_j. As such, the embedded transition kernel from active level g is

$$P_j(g,h) \;=\; q_{j,g}(h-g)\,c_j(h), \quad h \in \{g+1,\dots,G_j-1\}; \quad P_j(g,G_j) \;=\; \bar{q}_{j,g}(G_j-g), \tag{13}$$

and

$$P_j(g,\star) \;=\; \sum_{h=g+1}^{G_j-1} q_{j,g}(h-g)\,\big(1-c_j(h)\big) \;=\; 1-\bar{q}_{j,g}(G_j-g) - \sum_{h=g+1}^{G_j-1} q_{j,g}(h-g)\,c_j(h). \tag{14}$$

Payments are positive, so $P^{(j)}$ is strictly upper triangular and $(P^{(j)})^{G_j} = 0$.

Donor consumption and active projects. Let K_j^{don} denote the number of transitions required to reach an absorbing state, starting from arrival (the initial continuation draw is instantaneous), counted *per arrival*, with an arrival whose initial draw fails contributing zero:

$$K_j^{\mathrm{don}} \;:=\; \sum_{g=0}^{G_j-1} \mathbf{1}\{\text{the chain is active at level } g \text{ at some step}\}, \tag{15}$$

and let $\bar{K}_j^{\mathrm{don}} := \mathbb{E}[K_j^{\mathrm{don}}]$ be its mean. With probability $1-c_j(0)$, the initial draw fails and $K_j^{\mathrm{don}} = 0$; otherwise, the chain absorbs in at most G_j donor-consuming steps, so $\bar{K}_j^{\mathrm{don}}$ is finite. Note that $\bar{K}_j^{\mathrm{don}}$ is the plan's donor consumption per type-j arrival, so the total donor-match load imposed by type j is $\lambda_j \bar{K}_j^{\mathrm{don}}$.

Note that the absorption time for an admitted project can be either before or after its deadline. To decouple admission and capacity accounting from the real timeline and deadlines, our framework introduces a *virtual*

active set. In particular, when a project is admitted, it receives a *virtual label* that remains in the active set until the associated per-project chain absorbs at G_j or $\star$, regardless of whether the real deadline has passed. We call such a label *active* for as long as it remains in that set, *live* while it is also within its deadline, and *dead* once it is not. The value w_j is earned only if the virtual chain reaches G_j within real age D_j. If the virtual label is past its deadline (real age $> D_j$) when its chain absorbs, the absorption produces no real value but otherwise affects the admission and capacity accounting identically. In other words, both the Maintenance and Admission control modules care only about the virtual label and not the actual time. If a donor is routed to a virtual label whose real deadline has already passed, the interaction is a no-op in the real crowdfunding system and produces no value. The payment draw and the chain's embedded transition are simulated only for the virtual active-set bookkeeping.

With the continuation probabilities $c_j(\cdot)$ read from the LP optimum via (9), the following lemmas show that the chain follows the plan of the fluid LP exactly. In particular, the probability of absorption at the success state G_j equals ρ_j.

LEMMA 2 **(Visit and active recurrences from the LP).** *The entry probability $\hat{b}_j(g)$ and active probability $\hat{a}_j(g)$ of the survival chain of Section 4.1 satisfy $\hat{b}_j(0)=1,\ \hat{a}_j(0)=c_j(0)$, and for $g\in\{1,\dots,G_j-1\}$:*

$$\hat{b}_j(g) \;=\; \sum_{r=0}^{g-1} \hat{a}_j(r)\, q_{j,r}(g-r), \qquad \hat{a}_j(g) \;=\; c_j(g)\,\hat{b}_j(g). \tag{16}$$

Moreover, the absorption probability at G_j is

$$s_j \;:=\; \mathbb{P}[\textit{the chain absorbs at } G_j] \;=\; \sum_{r=0}^{G_j-1} \hat{a}_j(r)\,\bar{q}_{j,r}(G_j-r). \tag{17}$$

LEMMA 3 **(Embedded chain matches the plan's success rate and donor budget).** *With $c_j(\cdot)$ defined from the LP optimum as in (9), the chain's visit probabilities are the plan's, $\hat{b}_j=b_j$ and $\hat{a}_j=a_j$ of (10). Hence the chain's G_j-absorption probability equals the LP success rate: $s_j = \rho_j$, and its mean donor consumption equals the plan's donor budget per arrival:*

$$\bar{K}_j^{\mathrm{don}} \;=\; \sum_{g=0}^{G_j-1} a_j(g) \;=\; \frac{1}{\lambda_j}\sum_{g=0}^{G_j-1} M_j(g). \tag{18}$$

The chain thus reproduces the plan's success rate and donor budget. Whether the required donations arrive before the deadline, however, is a question that Maintenance cannot answer on its own; this is where Display comes in.

4.2. Display

Display is the module that delivers the deadline. Its commitment is a *floor*: it guarantees that every type-j project, from the moment it is admitted until it is absorbed at either of the two absorbing states (G_j or $\star$), is matched to donors at rate at least u_j. The mechanism is an assortment rule that realizes any vector of aggregate per-type rates exactly, as long as it is feasible given the donor supply; this in turn allows us to bound the absorption time by a phase-type random variable whose tail decays exponentially in u_j.

The exact-rate display distribution. Write $\bar{y}_j := \lambda_j \bar{K}_j^{\mathrm{don}}$ for the aggregate match rate the plan reserves for type j, and $m \in \mathbb{Z}_+^J$ for the current count of active projects by type. A *display distribution* at count m is a family $\mathcal{D}(m) = (\mathcal{D}_i(m))_{i\in\mathcal{I}}$ with $\mathcal{D}_i(m) \in \Delta(2^{\mathcal{J}})$, one assortment distribution per donor type, such that the induced aggregate match rate of every active project type equals its reserved rate exactly:

$$\sum_{i\in\mathcal{I}} \mu_i \, \mathbb{E}_{R\sim\mathcal{D}_i(m)}\big[\varphi_i(j \mid R)\big] \;=\; \bar{y}_j \, \mathbf{1}\{m_j \geq 1\} \qquad \forall j \in \mathcal{J}. \tag{19}$$

At each arrival of a type-i donor, the platform then draws $R \sim \mathcal{D}_i(m)$ and displays it.

LEMMA 4 **(Existence of an exact-rate display distribution).** *The reserved rate vector $\bar{\mathbf{y}} = (\lambda_j \bar{K}_j^{\mathrm{don}})_j$ lies in $\widehat{F}_+$: by Lemma 3 it is the plan's own match flow, $\bar{y}_j = \sum_g M_j(g)$, which the donor constraint of the LP places in $\widehat{F}_+$. Every $y \in \widehat{F}_+$ is implementable: there exists $\mathcal{D} = (\mathcal{D}_i)_i$ with $\mathcal{D}_i \in \Delta(2^{\mathcal{J}})$ whose induced aggregate match rates are y, as in (19). Since $\widehat{F}_+$ is downward closed, this holds in particular for every y obtained from $\bar{\mathbf{y}}$ by setting some coordinates to zero; so when $m_j = 0$, no assortment in the support of any $\mathcal{D}_i$ gives type j positive selection probability, and donor routing is always well-defined. Moreover, $\mathcal{D}$ is computable online in polynomial time by the constructive Carathéodory scheme of Aminian et al. (2026b), from I independent per-donor-type assortment optimizations.*

The deadline tail. Write $\tau_j(u_j)$ for an admitted type-j project's time to absorption under our display rule at floor rate u_j, and $\delta_j := \mathbb{P}[\tau_j(u_j) > D_j]$ for its deadline-miss probability. Conditional on being active, a project receives donor matches at rate at least $u_j > 0$ (chosen in Section 4.5). Consider a normalized CTMC whose embedded chain is the survival Markov chain built in Section 4.1, with total departure rate 1 in every state (as if donors were matched to this project as a Poisson process of rate 1). Let Ξ_j be the project's *effort requirement*, i.e., the time it takes this normalized CTMC to be absorbed; it is phase-type, being the sum of a random number K_j^{don} of unit-rate exponentials, with mean $\mathbb{E}[\Xi_j] = \bar{K}_j^{\mathrm{don}}$. Effort accrues at rate at least u_j, so $\tau_j(u_j) \leq \Xi_j/u_j$ pathwise, with equality at the constant rate u_j.

LEMMA 5 **(Deadline tail).** *Under a per-project floor rate u_j, the effort requirement Ξ_j is phase-type, and*

$$\delta_j \;\leq\; \mathbb{P}\big[\Xi_j > u_j D_j\big] \;=\; e^{-u_j D_j} \sum_{k=0}^{G_j-1} \sigma_k^{(j)} \, \frac{(u_j D_j)^k}{k!}, \qquad \sigma_k^{(j)} \;:=\; c_j(0)\, e_0^\top \big(P^{(j)}\big)^k \mathbf{1}, \tag{20}$$

where e_0 is the unit vector of level 0 and $\sigma_k^{(j)} = \mathbb{P}[K_j^{\mathrm{don}} > k]$ is the probability that the project is still active after k donation events, so $c_j(0) = \sigma_0^{(j)} \geq \sigma_1^{(j)} \geq \cdots \geq \sigma_{G_j-1}^{(j)} \geq 0$ and $\sigma_{G_j}^{(j)} = 0$. Equivalently, Ξ_j is a mixture of Erlangs,

$$\Xi_j \;\sim\; \big(1 - \sigma_0^{(j)}\big)\, \delta_0 \;+\; \sum_{\ell=1}^{G_j} \big(\sigma_{\ell-1}^{(j)} - \sigma_\ell^{(j)}\big)\, \mathrm{Erlang}(\ell, 1), \tag{21}$$

the weight on $\mathrm{Erlang}(\ell,1)$ being $\mathbb{P}[K_j^{\mathrm{don}} = \ell]$ and the atom at 0 the coin-failure mass $1 - c_j(0)$. Since $K_j^{\mathrm{don}} \leq G_j$, replacing every $\sigma_k^{(j)}$ by 1 — the skip-free worst case, in which the project passes through every level in turn — gives the simpler bound

$$\delta_j \;\leq\; \mathbb{P}\big[\mathrm{Poisson}(u_j D_j) < G_j\big] = e^{-u_j D_j} \sum_{k=0}^{G_j-1} \frac{(u_j D_j)^k}{k!} = O\big((u_j D_j)^{G_j-1} e^{-u_j D_j}\big), \tag{22}$$

the $O(\cdot)$ holding as $u_j D_j \to \infty$ at fixed G_j.

We note that the bounds above hold at any scale of the problem and decay exponentially fast as u_j increases with the market size. This tail bound converts Maintenance's success rate into the success-by-deadline bound that our guarantee uses.

COROLLARY 1 **(Success-by-deadline chance)**. *For any admitted type-j project, the probability that it is successfully funded by its deadline satisfies*

$$\mathbb{P}[\textit{project absorbs at } G_j \textit{ within real age } D_j] \geq (\rho_j - \delta_j)_+,$$

where $(x)_+ := \max(x, 0)$.

The gap of this module lies in the extent to which it can provide its floor guarantee: there is a limit on the number of projects of each type to which Display can simultaneously deliver its floor. Concretely, Display can deliver its floor to every active type-j project exactly when the count satisfies $(m_j u_j)_j \in \widehat{F}_+$. Maintenance only controls the load of active projects, keeping it critical, right at this limit; this fixes the shape of the count's distribution, and together with the box within which the count must stay, it determines the distribution exactly. Without such a box, the realized count would fluctuate with random project arrivals and cause problems for Display whenever it falls outside the feasible region. The next module, Admission control, converts the ex-ante guarantee of Maintenance into the ex-post guarantee that Display requires.

4.3. Admission control

The last part of the Triad is Admission control, which converts the ex-ante promise into an ex-post one and completes the loop. The rule has two parts. First, restrict the feasible region to the largest box under $\bar{y}_j/u_j$: admit an arriving type-j project if and only if the type-j count stays at or below $K_j^{\square}$. Second, start the system in stationarity by seeding phantom projects at time zero with an appropriate starting state, so every arrival faces the same admission probability from the first instant. In short, the count behaves as a critically loaded shared-server queue whose stationary count distribution is uniform over the box. An arrival is blocked exactly when the count sits at the boundary, which happens with probability $1/(K_j^{\square}+1)$[8]. In the following, we explain this in detail.

The admission region. The floor rate $u = (u_j)_j$ is a free positive parameter throughout this section; Section 6 chooses it by a search over the caps. The reserved aggregate rate $\bar{y}_j = \lambda_j \bar{K}_j^{\mathrm{don}}$ — the plan's own match flow for type j, by Lemma 3 — is shared equally among the active type-j labels, so a label is served at rate $\bar{y}_j/m_j$, equal to u_j at $m_j = \bar{y}_j/u_j$ and larger for smaller counts. Display carries out this even split, through its exact-rate assortment draws (Section 4.2). The admission region is the *commitment box* $\mathcal{K}^{\square} := \{n \in \mathbb{Z}_+^J : \ n_j \leq K_j^{\square} \text{ for every } j\}$ with buffer sizes

$$K_j^{\square} := \lfloor \bar{y}_j/u_j \rfloor, \qquad \text{so the shared rate never drops below } \ \bar{y}_j/K_j^{\square} \geq u_j. \tag{23}$$

[8] Remark 8 compares this with dedicating a constant rate u_j to each project.

The box $\mathcal{K}^{\square}$ is downward closed, keeps the per-project rate above the floor, and makes the count distribution factorize across types.

The virtual count process $m(t)$ remains a pure function of the virtual chains and is decoupled from real-deadline accounting.

The count $m(t)$ is thus governed by the virtual sojourn times $\tau_j(u_j)$, independent of D_j, so the product form below utilizes the mean effort $\mathbb{E}[\Xi_j] = \bar{K}_j^{\mathrm{don}}$. The real value can then be bounded from below using Corollary 1.

Having discussed all the elements of our Triad Ensemble, we now present our detailed end-to-end algorithmic framework, Curate-Share.

4.4. Algorithmic framework: End-to-End

First, let us define the following distribution, which plays a central role in our algorithmic design framework and analysis.

DEFINITION 1 (BALANCED-SHARING CONFIGURATION DISTRIBUTION). Write $x = (x_{j,g})_{j\in\mathcal{J},\,0\le g<G_j}$ for the *configuration*: $x_{j,g}$ is the number of active type-j labels currently at level g, and $n_j = \sum_g x_{j,g}$ is the type-j count. For a downward-closed admission region $\mathcal{F}$, the balanced-sharing configuration distribution is

$$\pi(x) = \frac{1}{Z}\prod_{j\in\mathcal{J}} \varrho_j^{\,n_j}\, n_j! \prod_{g=0}^{G_j-1} \frac{1}{x_{j,g}!}\, p_{j,g}^{\,x_{j,g}}, \qquad p_{j,g} := \frac{a_j(g)}{\bar{K}_j^{\mathrm{don}}}, \qquad \varrho_j := \frac{\lambda_j \bar{K}_j^{\mathrm{don}}}{\bar{y}_j}, \tag{24}$$

where $Z := \sum_{x:\, n(x)\in\mathcal{F}} \prod_{j\in\mathcal{J}} \varrho_j^{\,n_j}\, n_j! \prod_g p_{j,g}^{\,x_{j,g}}/x_{j,g}!$ normalizes π over the configurations whose count vector lies in $\mathcal{F}$. Summing over phases at fixed counts, the count distribution is the truncated product geometric

$$\mathbb{P}[N = n] = \frac{1}{Z}\prod_{j\in\mathcal{J}} \varrho_j^{\,n_j}, \qquad n \in \mathcal{F}, \tag{25}$$

and, given $N_j = n_j$, the level counts are $\mathrm{Multinomial}\big(n_j;\, p_{j,0},\dots,p_{j,G_j-1}\big)$. Under the exactly balanced donor rate $\bar{y}_j = \lambda_j \bar{K}_j^{\mathrm{don}}$, we have $\varrho_j = 1$, and (25) is then *uniform* on $\mathcal{F}$.

We now explain the initialization steps our algorithm takes.

Initialization Stage.

(I1) Solve the fluid program (P) of Section 3, or its fairness-constrained variant (P_F), and compute the continuation probabilities $c_j(\cdot)$ (9), the reserved rates $\bar{y}_j = \lambda_j \bar{K}_j^{\mathrm{don}}$, and the buffers $K_j^{\square} = \lfloor \bar{y}_j/u_j \rfloor$ (23).

(I2) Sample the virtual count $m^{(0)}$ from the stationary distribution π of the balanced-sharing count process for $\mathcal{F} = \mathcal{K}^{\square}$ — the distribution of Definition 1, which depends on the per-project chains only through the mean effort $\mathbb{E}[\Xi_j] = \bar{K}_j^{\mathrm{don}}$.

(I3) For each of the $m_j^{(0)}$ initial virtual labels of type j, sample its current level g from the *equilibrium phase distribution of an active label*

$$\pi_{\mathrm{eq}}^{j}(g) \;=\; \frac{a_j(g)}{\sum_{r=0}^{G_j-1} a_j(r)} \;=\; \frac{a_j(g)}{\mathbb{E}[K_j^{\mathrm{don}}]}, \qquad g \in \{0,\dots,G_j-1\}, \tag{26}$$

which is proportional to the expected time the survival chain spends in each active level (i.e., to the visit probability $a_j(\cdot)$).

REMARK 6. Note that the virtual active-set dynamics do not depend on ages; as such, initial virtual labels sampled by (I2)–(I3) inherit only a phase $g \sim \pi^j_{\mathrm{eq}}$, not an age. Real ages are assigned only at real-arrival events (age $= 0$ at arrival) and are used solely for value-accounting purposes. Additionally, the initial virtual labels never generate real value and do not appear in the deadline check; they exist only to give $m(0)$ its stationary distribution.

The initialization procedure above realizes the stationary count and equilibrium phase marginals, as we prove in Theorem 1. Note that the real project inventory starts empty, and the virtual initialization is what gives every arrival the stationary admission probability from the first instant. With this in mind, we next state the online steps of our algorithm:

Online Dynamics. Starting from the initial state above, the algorithm applies the following rules.

(O1) *Project arrival.* On a type-j arrival at time t with current count m, if $m_j < K^{\square}_j$, admit the project: a new virtual label of type j is added, and its initial level is determined by the initial continuation coin of Section 4.1, i.e., with probability $c_j(0)$ it enters active level 0, and with probability $1 - c_j(0)$ it transitions immediately to $\star$ and the slot is freed. Otherwise, reject the arrival.

(O2) *Donor arrival.* On a type-i donor arrival at time t with current count $m \neq 0$:

- The algorithm maintains the exact-rate display distribution $\mathcal{D}(m) = (\mathcal{D}_{i'}(m))_{i' \in \mathcal{I}}$ of (19), which realizes, for each nonempty type j (with $m_j > 0$), the reserved aggregate donor match rate $\bar{y}_j = \lambda_j \bar{K}^{\mathrm{don}}_j$ (feasible and realizable by Lemma 4).
- Sample $R \sim \mathcal{D}_i(m)$ and display it to the donor.
- The donor selects $j \in R \cup \{\varnothing\}$ with probability $\varphi_i(j \mid R)$.
- If the donor selects $\varnothing$, the donor event ends: no payment is drawn and no virtual transition occurs.
- If type j is selected (implying $m_j > 0$), route the donor to a uniformly random type-j *virtual label* among the m_j active labels. Let g be its current active level. If the selected virtual label is past its real deadline, the platform discards the donor in the real system (no value is earned); this is feasible because discarding a donor is equivalent to inducing the outside option (see the alternative rule (O2′) below). In either case, the virtual chain transition below takes place, driven either by the donor's payment or by a simulated payment, per the convention of Section 4.1.
- If a virtual label has been selected (i.e., the donor event has not terminated above), apply the embedded transition rule of the survival chain of Section 4.1: draw $p \sim q_{j,g}(\cdot)$; let $h = \min(g + p, G_j)$. If $h = G_j$, the virtual label absorbs at G_j (a success, which earns value w_j if and only if the label's real age is at most D_j). If $h < G_j$, the label survives into active level h with probability $c_j(h)$; otherwise, it transitions immediately to $\star$ without consuming a further donor.

(O3) *Departure.* A virtual label leaves the active set when its chain absorbs at G_j or $\star$ (regardless of its deadline).

REMARK 7 (ALTERNATIVE ROUTING WITHOUT DISCARDING). The rule (O2) above routes all donors through the assortment mixture $\mathcal{D}(m)$ on the *virtual* active set; if the routed virtual label is past its real deadline, the donor is discarded. To avoid such donor discarding, a first approach is to route the donor only to a live project, uniformly at random, once the type is chosen. We think that would work, and would probably also help with the deadline and lower the count process in the first-order stochastic sense, and hence reduce blocking as well. The intuition is that each live project may now receive a higher rate of donor attention, since the donors previously discarded when the chosen project was dead are no longer wasted. However, this complicates the analysis: the count is then no longer a process that our analysis can characterize. With this in mind, the following alternative also avoids donor discards, while keeping the system's behavior from the live projects' perspective equivalent to (O2), and so retains the tractability and the simplification of the deadline and count analysis we already have.

(O2′) Let $\hat{m}_j(t) \le m_j(t)$ be the number of type-j virtual labels not past their real deadline at time t. Draw $R \sim \mathcal{D}_i(m(t))$ as in (O2), then apply the sub-assortment sampling of Feng et al. (2024) a second time (or the constructive Carathéodory scheme of Aminian et al. 2026b) to display an $R' \subseteq R$ under which type j is selected at aggregate rate $\bar{y}_j \hat{m}_j(t)/m_j(t)$, and match a donor who selects j uniformly among the $\hat{m}_j(t)$ live type-j labels. Dead labels receive no donors, but their chains are still simulated for the count's bookkeeping purposes (Section 4.1).

From the perspective of live projects, (O2′) is exactly equivalent to running (O2) and discarding the donor whenever it selects a dead label. Under (O2), each of the m_j labels receives rate $\bar{y}_j/m_j$, so the live ones receive $\bar{y}_j\hat{m}_j/m_j$ in total; under (O2′), type j receives $\bar{y}_j\hat{m}_j/m_j$ and splits it uniformly among its $\hat{m}_j$ live labels, again $\bar{y}_j/m_j$ each. Every live label sees the same donation rate at every state, and no donor is spent on a dead one.

The following figure illustrates the behavior of each of the three modules.

4.5. The guarantee

In the following we put together the properties that each module of our Triad provides and construct our main guarantee. At a high level, if everything went ideally, the arrivals of each project type would follow the guidance of the fluid LP exactly, and we could match its upper bound. However, two things can go wrong: (i) Admission control may block an arrival, in which case its value is lost; or (ii) even if the project is admitted, its absorption time (which matters only if it absorbs at G_j) may exceed its deadline, in which case again no real value is gained. Admission control bounds the impact of the former, and Display bounds that of the latter, as we detail below.

We first recall that the guarantees below are stated against the driving program, i.e., they hold whether we consider the unconstrained problem with (P) as the fluid relaxation or the fairness-constrained one with (P_F). We highlight that our guarantees are bi-criteria: our policies achieve the same factor for both the objective and the floor constraints.

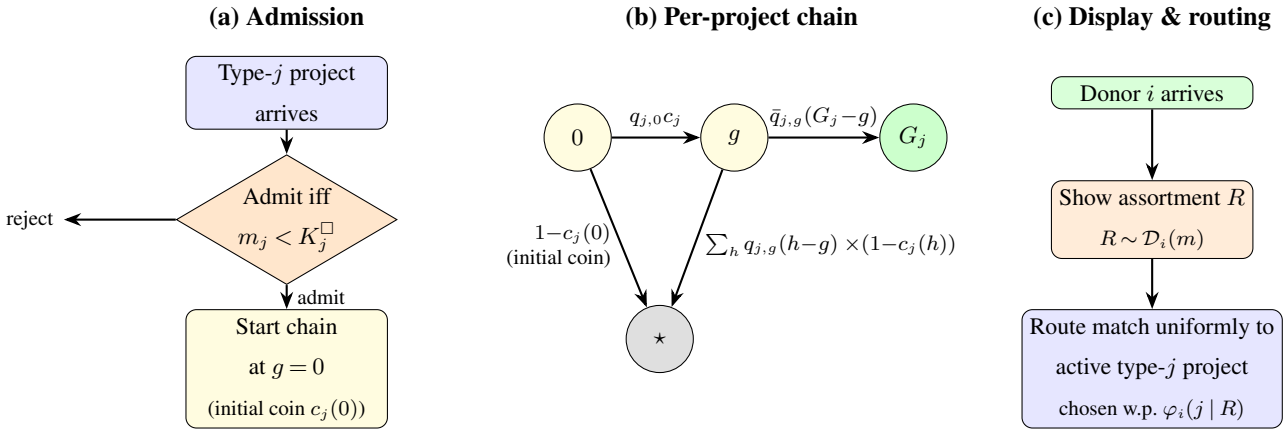


Figure 3 The algorithm in three steps. *(a) Admission:* an arriving type-j project is admitted if and only if the type-j count stays within the commitment box $\mathcal{K}^{\square}$, i.e., $m_j < K_j^{\square}$; on admission a per-project chain is started at level $g=0$ after the initial continuation coin $c_j(0)$. *(b) Per-project chain:* from active level g a donor tick draws a payment $p \sim q_{j,g}$ and moves to $h=\min(g+p, G_j)$; the project survives into active level h via the continuation probability $c_j(h)$ (with embedded kernel $P_j(g,h)=q_{j,g}(h-g)\,c_j(h)$), reaches the goal G_j through the overshoot tail $\bar{q}_{j,g}(G_j-g)$, or is absorbed at the null state $\star$ with probability $\sum_{h<G_j} q_{j,g}(h-g)\,(1-c_j(h))$, the destination failure probabilities summed over destinations, as in (14). *(c) Display & routing:* on each donor arrival the platform shows the assortment R drawn from the exact-rate display distribution $\mathcal{D}_i(m)$ of the arriving donor's type, and a type-i donor selects type j with probability $\varphi_i(j \mid R)$, routing a match to a uniformly chosen active type-j project.

The count and its blocking. Fix a type j. Its survival chain (Section 4.1) is a monotone absorbing phase process on active levels $g \in \{0,\dots,G_j-1\}$: a donor tick at level g advances the project to a higher level $h>g$ with probability $P_{gh}^{(j)}$, or absorbs it (success or failure) with probability $q_g^{(j)} = 1-\sum_h P_{gh}^{(j)}$. Recall that $a_j(g)$ is the probability that an arrival is ever active at level g, and the following recursion restates (11):

$$a_j(0)=c_j(0), \qquad a_j(h)=\sum_{g<h} a_j(g)\,P_{gh}^{(j)} \;\; (h\geq 1), \qquad \sum_g a_j(g)=\bar{K}_j^{\mathrm{don}} \tag{27}$$

(equivalently $a^{(j)} = c_j(0)\,e_0^\top (I-P^{(j)})^{-1}$: $c_j(0)\,e_0^\top (P^{(j)})^k$ is the chance of being active at each level after exactly k donations, and a level is visited at most once, so these add to $a^{(j)} = c_j(0)\,e_0^\top \sum_{k\geq 0}(P^{(j)})^k$; the sum is finite since $(P^{(j)})^{G_j}=0$, and equals $(I-P^{(j)})^{-1}$). This simple form holds because each level is visited at most once, and $K_j^{\mathrm{don}} = \sum_g \mathbf{1}\{\text{active at } g\}$ has mean $\sum_g a_j(g)$. Each active visit consumes exactly one donation, so $\bar{K}_j^{\mathrm{don}}$ is the mean number of donations *per arrival* and the offered workload is $\lambda_j \bar{K}_j^{\mathrm{don}}$.

THEOREM 1 **(Insensitive product form).** *Consider running our algorithm of Section 4.4 with admission to a finite downward-closed region $\mathcal{F}$. Then the virtual configuration process is a positive recurrent Markov chain whose stationary distribution is that of Definition 1. Its count marginal (25) depends on the funding dynamics* only through the mean $\bar{K}_j^{\mathrm{don}}$ *(insensitivity).*[9]

[9] This is an instance of the insensitivity of processor-sharing networks (Kelly, 1979; Whittle, 1985); we prove it directly by partial balance, which is what transfers to the enlarged chain of Section EC.2.2.

Write α_j^{block} for the *blocking probability*, the stationary probability that an arriving type-j project is rejected, and $1-\alpha_j^{\text{block}}$ for the corresponding *admission probability*. Admission control is the only step at which the policy rejects, so α_j^{block} is the whole of the admission loss. At criticality, the resulting uniform distribution makes the blocking probability a purely geometric quantity—the fraction of $\mathcal{F}$ on its type-j upper boundary. By PASTA a type-j arrival is blocked exactly when $n+e_j \notin \mathcal{F}$, and downward closure gives a bijection between the blocking states and the states in $\{n \in \mathcal{F} : n_j = 0\}$, so

$$\alpha_j^{\text{block}} = \frac{\left|\{n \in \mathcal{F} : n + e_j \notin \mathcal{F}\}\right|}{|\mathcal{F}|} = \mathbb{P}[N_j = 0]. \tag{28}$$

For the per-type box $\mathcal{F} = \mathcal{K}^{\square}$, this is $1/(K_j^{\square}+1)$; for a shared-budget simplex $\mathcal{F} = \{n : \sum_j n_j \le K\}$, it is $|\mathcal{J}|/(K+|\mathcal{J}|)$. Note that both of these are $\Theta(1/K)$, the canonical boundary-to-volume ratio. In this paper, we use the box $\mathcal{K}^{\square}$ as it fits inside the donor-feasible region $\{n : (n_j u_j)_j \in \widehat{F}_+\}$.

LEMMA 6 **(Critical load, uniformity, and blocking)**. *The reserved rate is $\bar{y}_j = \lambda_j \bar{K}_j^{\text{don}}$ by definition, so the type-j demand load and supply load coincide and $\varrho_j = 1$ identically — the system is critically loaded in every type, with no condition on the instance. The count distribution (25) of Definition 1 is therefore uniform on the commitment box $\mathcal{K}^{\square}$, and a type-j arrival is blocked exactly when the type-j count sits at the cap, so*

$$\alpha_j^{\text{block}} = \mathbb{P}[N_j = K_j^{\square}] = \frac{1}{K_j^{\square}+1}. \tag{29}$$

Blocking falls as the box caps $K_j^{\square}$ grow: $1/(K_j^{\square}+1)$ decreases in $K_j^{\square}$, so under the same sharing rule a larger cap blocks less, and what prevents us from enlarging it is the floor — by (23), $K_j^{\square}$ is the largest cap at which the shared rate still clears u_j.

THEOREM 2 **(Per-type success rate)**. *Let π be our Curate-Share policy described in Section 4.4. Then, for every type $j \in \mathcal{J}$, a type-j project arriving at any point in time is successfully funded with a probability ρ_j^{ALG} that does not depend on its arrival time, and*

$$\rho_j^{\text{ALG}} \;\ge\; (1-\alpha_j^{\text{block}})\,(\rho_j - \delta_j)_+; \tag{30}$$

consequently, the type-j success rate converges: $S_j^{\pi,T} \to \lambda_j\, \rho_j^{\text{ALG}}$ as $T \to \infty$.

COROLLARY 2. *Let π be our Curate-Share policy described in Section 4.4. Then, the following holds:*

$$J(\pi) \;\ge\; \sum_{j\in\mathcal{J}} \lambda_j\, w_j\, (1-\alpha_j^{\text{block}})\,(\rho_j - \delta_j)_+, \tag{31}$$

and consequently

$$J(\pi) \;\ge\; \Big(\min_j (1-\alpha_j^{\text{block}})\,(1-\delta_j/\rho_j)_+\Big)\cdot \text{OPT}_{\text{LP}}. \tag{32}$$

Hence, $J(\pi) \ge \text{CR}\cdot\text{OPT}_{\text{LP}}$ with $\text{CR} = \min_j (1-\alpha_j^{\text{block}})(1-\delta_j/\rho_j)_+$, and for every $j \in \mathcal{P}$, the per-arrival success rate ρ_j^{ALG} of Theorem 2 obeys $\rho_j^{\text{ALG}} \ge (1-\alpha_j^{\text{block}})(\rho_j-\delta_j)_+ \ge \text{CR}\cdot\rho_j \ge \text{CR}\cdot\beta_j$, using $\rho_j \ge \beta_j > 0$. The policy is thus simultaneously CR*-competitive and* CR*-approximately fair, and the fairness holds project by project: a protected project arriving at any point in time is funded with probability at least $\text{CR}\cdot\beta_j$.*

REMARK 8 (DEDICATED RATES INSTEAD OF SHARING). Display could instead give each active type-j project exactly its dedicated floor rate u_j, regardless of how empty that type is, on the same box $\mathcal{K}^{\square}$ (the Triad with dedicated, constant rates, which we call Curate-Dedicate); the type-j count is then an Erlang loss system, still insensitive, and Theorem 2 and Corollary 2 hold verbatim with $\alpha_j^{\mathrm{block}}$ read as its blocking probability. Dedicated rates also decouple the labels: each label is served at its own constant rate u_j, so conditional on admission the per-label chains are independent CTMCs and per-label success can be evaluated in closed form — the loss-network reduction the numerics of Section 6 run on. Sharing does not decouple them, since a label's rate $\bar{y}_j/m_j$ depends on how many others are active. However, the stationary count is then a truncated Poisson distribution with its mean at the boundary rather than uniform, so the boundary mass, and with it the blocking probability, is $\Theta(1/\sqrt{K_j^{\square}})$ in place of $1/(K_j^{\square}+1)$, quadratically worse. Only the rate of Theorem 3 does not transfer: under its scaling Curate-Dedicate reaches $1-\tilde{O}(1/\sqrt{\kappa})$. Indeed, at the clock $u_j=\log\kappa/D_j$ of Theorem 3 every admitted project is served at exactly u_j, so (22) gives the same $\delta_j=O\big((\log\kappa)^{G_j-1}/\kappa\big)$ as there. Its box $\mathcal{K}^{\square}$ has $K_j^{\square}=\lfloor\bar{y}_j/u_j\rfloor=\Theta(\kappa/\log\kappa)$ with the Poisson mean $\bar{y}_j/u_j$ at the boundary, so the blocking above is $\alpha_j^{\mathrm{block}}=\Theta(1/\sqrt{K_j^{\square}})=\tilde{O}(1/\sqrt{\kappa})$, and (32) gives $\mathrm{CR}\geq\min_j(1-\alpha_j^{\mathrm{block}}-\delta_j/\rho_j)=1-\tilde{O}(1/\sqrt{\kappa})$, as ρ_j remains constant under Definition 2.

Having stated our main technical theorem, we now explain how different terms in it interact with each other. In particular, the two factors $\alpha_j^{\mathrm{block}}$ and δ_j pull against each other through our free parameter u_j. A larger floor rate speeds projects up (inducing smaller δ_j) but shrinks the buffer $K_j^{\square}=\lfloor\bar{y}_j/u_j\rfloor$ (implying larger blocking); note that with $\bar{y}_j=\lambda_j\bar{K}_j^{\mathrm{don}}$ and $1/(K_j^{\square}+1)\leq u_j/\bar{y}_j$, we have

$$\alpha_j^{\mathrm{block}}=1/(K_j^{\square}+1)\;\leq\;\frac{u_j}{\lambda_j\bar{K}_j^{\mathrm{don}}},\qquad \delta_j\;\leq\;e^{-u_jD_j}\sum_{k=0}^{G_j-1}\sigma_k^{(j)}\,\frac{(u_jD_j)^k}{k!}=O\big((u_jD_j)^{G_j-1}e^{-u_jD_j}\big). \tag{33}$$

Given a fixed instance of the problem, one can always find the cap $K_j^{\square}$, and its corresponding rate $u_j=\bar{y}_j/K_j^{\square}$, that maximizes the objective via a simple one-dimensional search over the integers. In the following, we focus on a canonical market scaling regime and discuss how to balance these two terms to obtain a clean approximation factor.

DEFINITION 2 (CANONICAL LARGE-MARKET SCALING). The *canonical large-market scaling* is the regime in which the arrival rates λ_j,μ_i grow while the goals G_j, deadlines D_j, and per-project transition chains $P^{(j)}$ stay fixed, and the policy is driven by a fixed normalized optimum, scaled with the market — the fluid program (P) is homogeneous of degree one in (λ,μ), so such a normalized optimal solution exists and makes $c_j(\cdot)$, $\bar{K}_j^{\mathrm{don}}$ and ρ_j scale-invariant even when the optimum is not unique. We write κ for the common arrival scale of the market, so that $(\lambda,\mu)=\kappa\,(\lambda^\circ,\mu^\circ)$ where $(\lambda^\circ,\mu^\circ)$ are the arrival vectors of the normalized instance, and D for a representative deadline.

Keeping this scaling in mind, first note that the blocking probability is $\Theta(u_j)$, while the deadline miss behaves as $e^{-\Theta(u_j)}$. This means that the main bottleneck is the blocking that comes from Admission control: a slight increase in the floor rate drives the deadline miss down exponentially fast, while the blocking responds to

it only linearly. Under this scaling, $K_j^{\square} = \Theta(\kappa)$ at any fixed clock and $\Theta(\kappa/\log\kappa)$ at the clock of Theorem 3, so the blocking written in terms of $K_j^{\square}$ and the competitive ratio written in terms of κ agree up to the logarithm that $\tilde{O}$ absorbs.

THEOREM 3 **(Large Market behavior of the bi-criteria competitive factor)**. *In the regime of Definition 2 where the arrival rates λ_j, μ_i scale while the goals G_j, deadlines D_j, and DTMCs $P^{(j)}$ stay fixed, we let the floor rate u_j grow as $\Theta(\log\kappa)$, i.e., $u_j = \log\kappa/D_j$. Then, for every j,*

$$\alpha_j^{\text{block}} = O\Big(\frac{\log\kappa}{\kappa}\Big), \qquad \delta_j = O\Big(\frac{(\log\kappa)^{G_j-1}}{\kappa}\Big),$$

both of which are $\tilde{O}(1/\kappa)$; hence, for our Curate-Share policy π of Section 4.4,

$$\boxed{J(\pi) \;\geq\; \big(1-\tilde{O}(1/\kappa)\big)\,\mathrm{OPT}_{\mathrm{LP}}}.$$

4.6. The upper bound

The following proposition states that our achievable bounds are tight up to polylogarithmic factors: in particular, any feasible policy loses an $\Omega(1/\kappa)$ fraction relative to the fluid benchmark.

PROPOSITION 2 **(No policy beats $1/\kappa$)**. *Consider the single-type instance: $\mathcal{J} = \{1\}$, $G = 1$ with unit payments, deterministic deadline D, project arrivals Poisson(λ), a single donor type arriving Poisson(μ) who donates to the displayed project, and balanced rates $\lambda = \mu$,*[10] *so that $\mathrm{OPT}_{\mathrm{LP}} = \mu$. Then every policy π — including every* offline, clairvoyant *policy — satisfies*

$$J(\pi) \;\leq\; \Big(1 - \frac{1}{1+\mu D}\Big)\mathrm{OPT}_{\mathrm{LP}}\,,$$

with equality attained by first-in-first-out service. Hence every policy loses $\Omega(1/\kappa)$ on this instance.

Theorem 3 and Proposition 2 together bracket the achievable rate: our policy π attains $1-\tilde{O}(1/\kappa)$, and no policy — online or offline — attains better than $1-\Omega(1/\kappa)$ against the fluid benchmark on a deterministic-deadline instance. Whether the polylogarithmic factor separating the two is necessary remains open.

The guarantee is now as tight as the donor side allows, and the ceiling, $\mathrm{OPT}_{\mathrm{LP}}$, is what a fixed supply of donor attention can fund. In many cases, though, platforms are not confined to that supply. The next section gives the platform a budget of its own money and asks the same question again.

5. Sponsor capital

Our core model in Section 2 treats donor attention as the platform's only resource. In the real world, however, donation-based crowdfunding platforms rarely run on traditional retail donors alone: the platforms routinely receive large donation amounts from governments, NGOs and non-profit foundations, and even from corporate philanthropy, and can use these budgets to advance the projects they receive at different stages of their

[10] Balance is the natural case to state: at $\lambda = \mu$, the level-crossing equation admits the constant solution $g \equiv \mu p_0$, which is what makes the loss exactly computable. We do not claim the balanced instance is extremal within the family; a single instance suffices for the upper bound.

progress. Section EC.2 incorporates this lever: the platform spends capital at a long-run rate of at most B, charged like donations only when a project succeeds, so the "stake" in finishing a project depends on how much sponsor capital has already been spent on it, and the state must also carry that capital. On this enlarged state, a fluid program again upper-bounds every budget-feasible policy (Theorem EC.1), and Curate-Share run off its optimum, with the display and admission rules unchanged, attains it up to the same factor, $J(\pi^{\mathrm{enl}}) \geq (1-\tilde{O}(1/\kappa))\,\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B) \geq (1-\tilde{O}(1/\kappa))\,\mathrm{OPT}_{\mathrm{LP}}$ (Theorem EC.2). Section 6.7 puts numbers on this lever.

6. Numerical study and managerial insights

The theoretical guarantees of Section 4.5 and Section EC.2 hold for every instance of the model class we study (Section 2). In this section, we put on a practical lens and evaluate the actual performance of our policy framework in various aspects. We calibrate every primitive of the model to the public DonorsChoose data (KDD Cup, 2014; 201,419 projects and 1.16 million donations around the 2013 calibration year), implement the algorithm of Section 4 as specified, and compare its realized value rate with the certified optimum of the fluid LP of Section 3.1.

Our main finding is that Curate-Share, the policy we recommend, at its searched admission caps reaches 99.7% of the LP optimum under the throughput weighting and outperforms every greedy heuristic we test by 2.5 points there, 7.7 under revenue and 11.7 under the mission weighting. Curate-Dedicate, the constant-per-project-rate counterpart of Curate-Share (Remark 8), realizes 86.8%, whereas Curate-Share on that same box realizes 97.4%.

6.1. Data, calibration, and implementation

In this subsection, we explain how every primitive of the model is estimated from the public DonorsChoose data. We calibrate on the $131{,}304$ projects posted in 2013. Project types are the $J = 42$ cells of primary focus area, highest-poverty indicator and goal tercile, donor types are the U.S. states plus an out-of-coverage bucket ($I = 52$), deadlines are $D_j \equiv 120$ days, payment distributions are estimated per type and level quartile, and donor choice is multinomial logit with a full-display conversion rate of $c = 0.90$. The protected set is the 21 highest-poverty types, 54% of arrivals, and value is weighted in three ways: throughput, revenue, and a mission weighting by students reached. With $w_j \equiv 1$, the fluid optimum funds 77.6% of arriving projects, against the platform's realized 70.4% in the 2013 data. Section EC.4.1 gives the calibration in full, the computation of the benchmark and the implementation of the policy.

6.2. Headline performance against greedy baselines

Figure 4 compares our policies with greedy heuristics. All of these heuristics admit every project and display every live type, and they differ only in how a donor's selected type is routed to a project. The rules are uniform routing, oldest-first (FCFS), newest-first (LCFS), expiring-first, and *almost-funded-first* (AFF), which routes to the live project of the selected type with the smallest remaining amount to its goal. Under this rule only the routed project gains level, so whenever it completes or expires, the type's other live projects are tied at level zero. The admit-all baselines stay stable despite the critical load, because a project leaves at its deadline

whether or not it is funded, so the live count is self-limiting. A popularity baseline (top-12) instead restricts display to the top twelve types by aggregate attraction.

The canonical box caps, $K_j^{\square} = \lfloor \bar{y}_j/u_j \rfloor = \lfloor \lambda_j D_j/2 \rfloor \in [87, 1793]$ at the canonical clock $u_j = 2\bar{K}_j^{\mathrm{don}}/D_j$, at which a project's mean donor effort $\bar{K}_j^{\mathrm{don}}$ takes half its deadline, are built from λ and D alone. The stationary count has the exact uniform product form, and we simulate the policy, estimating by cohort, i.e., from the funded fraction of the projects admitted during the measurement window, as Section EC.4.1 describes. At the canonical box, Curate-Share realizes 0.97, 0.98 and 0.99 under throughput, revenue and mission weights, respectively, and at its searched caps 0.997, 0.996 and 0.995. At its searched caps, which run from 4 to 871, Curate-Share's 0.997 is far above what its smallest cap would suggest, because the ratio is value-weighted over all types and most of the value sits in types with much larger caps, whereas the bound (32) takes the minimum over types.

On the calibrated instance Curate-Dedicate at the canonical clock realizes

$$\frac{\hat{J}}{\mathrm{OPT}_{\mathrm{LP}}} = 0.868 \pm 0.001\ (w \equiv 1), \qquad 0.886 \pm 0.001\ (w_j = G_j), \qquad 0.885 \pm 0.001\ (w_j = \text{students}).$$

Under the throughput weighting, almost-funded-first reaches 0.972 of $\mathrm{OPT}_{\mathrm{LP}}$ (0.919 and 0.879 under revenue and mission weights). Expiring-first coincides with FCFS in any calibration of this model, because the deadline $D_j = 120$ days is common to all types, so ordering by expiry is ordering by arrival. The two reach 0.86, as does LCFS, all three just below Curate-Dedicate at the canonical clock (0.868). Uniform routing reaches 0.74; like LCFS, it strands partial funding.

Restricting display to the top 12 types reaches 0.86 under throughput, above uniform routing, because a restricted assortment concentrates scarce donor attention instead of only shedding flow. Similarly, it reaches 0.84 under the revenue weighting but 0.60 under the mission weighting — a 26-point spread in performance depending on the weighting used, against 8–9 points for the full-display rules. Popularity display is therefore selectively destructive, and what it destroys is exactly the project types the platform cares the most about.

In Figure 4, a policy that collapses shows as a bar shrinking in place. Curate-Share at its searched caps leads the strongest baselines by 2.5, 7.7 and 11.7 points. The heuristics all underperform our policy, and we believe the main causes are that they either do not concentrate the donations enough (especially LCFS and uniform), put all their eggs into one basket (like AFF and FCFS) even though it might have been already late to save a project, or they concentrate them over projects that are not the priorities of the platform (e.g. the top-12 popularity display).

Pool a category's exposure. Based on our results, we recommend the following operating practices. The change with the highest value is in how the recommendation system allocates attention. Instead of giving each campaign its own fixed slice of exposure, the platform should give each *category* a fixed exposure budget, sized to its historical demand, and split that budget in real time among whichever of its campaigns are live. A campaign then gets more visibility exactly when few of its category peers are live, which is most of the time, and falls back to a baseline only when the category is crowded.

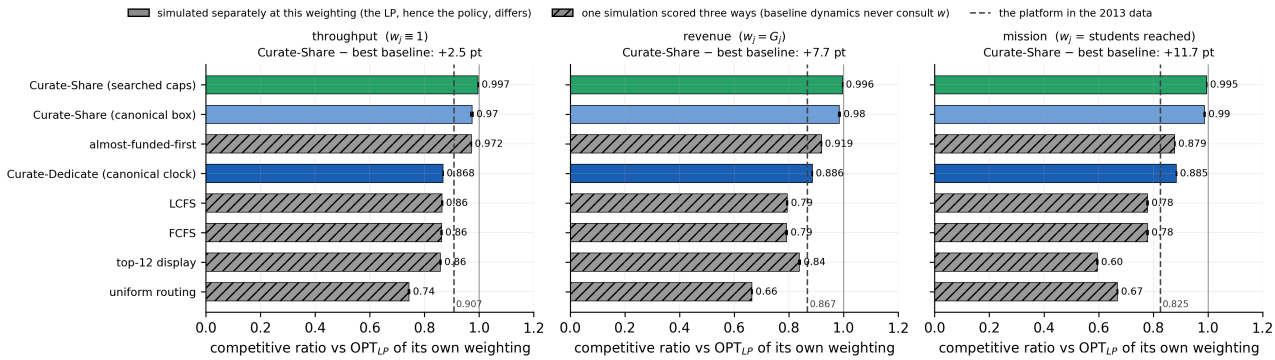


Figure 4 Service rules against admit-all baselines, all three weightings. Each panel scores every policy against the LP optimum of its own weighting. Solid bars are simulated separately at each weighting, because the LP, and so the plan the policy runs, differs by weighting. Hatched bars are one simulation scored three ways, which is exact here, since baseline dynamics never consult w and only the value assigned to the same sample path changes. Rows keep the order of the throughput panel across all panels. The dashed line is the platform's realized funding in the 2013 data, valued under each weighting.

6.3. The effect of scaling

On the calibrated instance, the realized value of Curate-Dedicate has an exact expression. Write $V_j := \mathbb{P}[\text{absorb at } G_j \text{ within real age } D_j]$ for the exact success-by-deadline probability of the chain of Section 4.1. Under Curate-Dedicate, on the box $\mathcal{K}^{\square}$, the per-type bound (31) sharpens to an identity, since the count is a product of independent truncated-Poisson coordinates and admission is independent of each admitted label's chain (Remark 8):

$$J(\pi) \;=\; \sum_{j\in\mathcal{J}} \lambda_j\, w_j\, (1-\alpha_j^{\mathrm{block}})\, V_j.$$

In Figure 5, the calibrated instance has $\min_j \bar{y}_j/u_j \approx 87$ at the canonical clock $u_j = 2\bar{K}_j^{\mathrm{don}}/D_j$, and the curve of Curate-Dedicate is evaluated exactly as $\sum_j \lambda_j w_j\,(1-\alpha_j^{\mathrm{block}})\,V_j/\mathrm{OPT}_{\mathrm{LP}}$. Both searched caps are searched at every scale, and their clocks grow like $\Theta(\log\kappa)$, so the deadline miss vanishes and the admission loss is what remains, quadratically larger under Curate-Dedicate (Remark 8).

Common scaling preserves every structural quantity of Section 3.2 (ρ_j, $c_j(\cdot)$ and $\bar{K}_j^{\mathrm{don}}$ are ratios of LP flows, all linear in κ), so Figure 5 isolates the two scale-dependent factors of (32), admission and the deadline miss. We draw two lessons.

(i) The canonical clock burden. With the canonical clock $u_j = 2\bar{K}_j^{\mathrm{don}}/D_j$, Curate-Dedicate's ratio rises only from 0.770 to 0.882 as $\min_j K_j^{\square}$ grows from 4 to 348 (not drawn in Figure 5). The admission factor improves at the $1 - O(1/\sqrt{K_j^{\square}})$ rate, but $u_j D_j$, the only quantity the deadline miss depends on, is scale-free under a clock that does not grow with the market. As such, as the market scales, the deadline miss dominates the loss, and saturates the performance to a constant below 1. Curate-Share faces the same issue under such a scale-free canonical clock rate, but converges to a better constant as projects get funded faster under this policy.

(ii) Searching for the best clock restores convergence. Optimizing each u_j by a per-type search, i.e., maximizing the exact product $(1-\alpha_j^{\mathrm{block}}(u))\cdot V_j(u)$, lifts the ratio to $0.844, 0.894, 0.922, 0.943, 0.958, 0.970, 0.978$

across the seven scales. Fitted over all seven scales, each doubling of market size multiplies the residual loss by about 0.73, so the loss scales as $\kappa^{-0.45}$, consistent with the $\Theta(1/\sqrt{K_j^{\square}})$ admission gap once the deadline factor no longer binds. Curate-Share's loss at its searched caps falls faster, as $\kappa^{-0.85}$. The objective is the type's simulated success per arrival, $K_j^{\square}/(K_j^{\square}+1)$ times the funded fraction of its admitted projects.

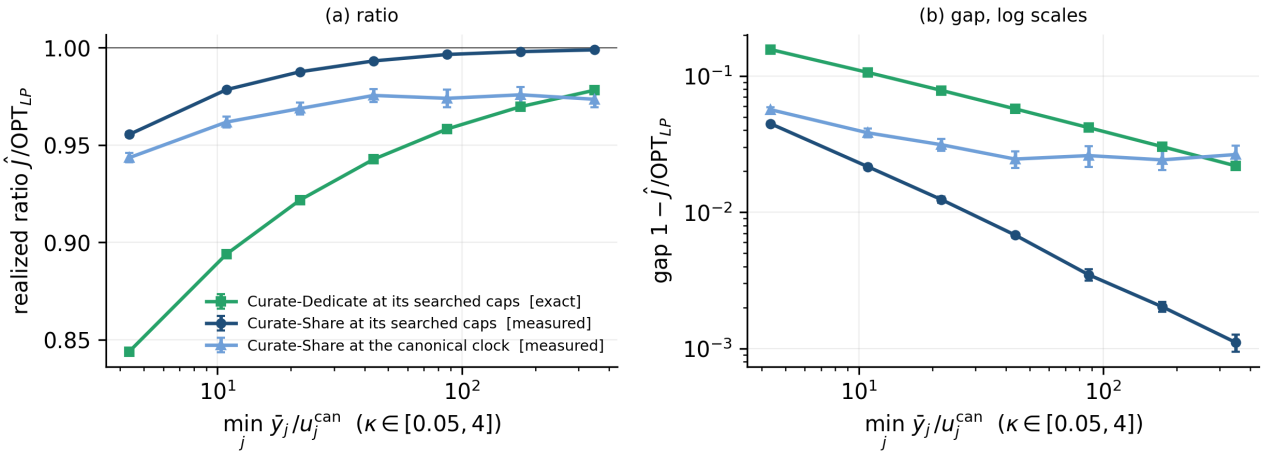


Figure 5 Realized ratio versus market scale. Common scaling $\kappa \cdot (\lambda, \mu)$ with $\kappa \in [0.05, 4]$; the calibrated instance is $\kappa = 1$. The three curves are Curate-Share at the canonical clock, Curate-Share at its searched caps, and Curate-Dedicate (Remark 8) at its own searched caps.

Figure 5 draws this expression for Curate-Dedicate at its own searched caps. The loss-network simulations of Curate-Dedicate agree with this expression to within 0.002 at every scale, and an independent donor-stream simulator that realizes V_j, instead of computing it, which can be done exactly only under Curate-Dedicate, agrees at the calibrated scale, 0.8683 ± 0.0010 against 0.8685.

The balanced-sharing refinement of Section 4.3 removes most of Curate-Dedicate's remaining gap. Pooling each type's donor stream and sharing it equally among its active projects turns the per-type count from $M/G/\infty$ into $M/G/1$, and the admission loss falls quadratically, from 8.2% to 1.1% at $\min_j \bar{y}_j/u_j = 87$, with the ratio of the two losses growing as $\sqrt{K_j^{\square}}$, to $15\times$ at $\kappa = 4$.

6.4. Donor scarcity

Donor scarcity separates the policies further, and it separates Curate-Dedicate even from almost-funded-first. When we rescale $\mu \to \phi\,\mu$ and re-solve the benchmark, Curate-Share at its searched caps holds 0.994–0.997 and Curate-Dedicate at its own searched caps 0.948–0.958 as ϕ falls to one third, while almost-funded-first, the strongest heuristic at every ϕ and weighting, falls to 0.897 and 0.897 at $\phi = 0.5$ and $\phi = 0.33$, respectively. Hence, Curate-Share then leads it by ten points, and Curate-Dedicate by five to six points (Table 2). Across a threefold contraction, Curate-Share loses 0.2 points and Curate-Dedicate one point, while almost-funded-first drops by 7.5 points; FCFS and uniform routing lose 40–50% of their ratio to the optimum, and top-12 display, which loses the most, 51%.

donor supply	Curate-Dedicate (searched caps)	Curate-Share (searched caps)	almost-funded-first
$\phi = 1$ (calibrated)	0.958	0.997	0.972
$\phi = 0.5$	0.956	0.996	0.897
$\phi = 0.33$	0.948	0.994	0.897

Table 2 Donor-scarcity stress test. Realized value over the LP optimum, $w \equiv 1$, at the calibration used here (10% outside share); $\mu \to \phi\,\mu$ with the benchmark re-solved at each ϕ.

6.5. Erroneous estimates

Here, we study the performance robustness of different policies, w.r.t. estimation errors on different parameters. In particular, we run the policies based on the plan derived from the calibrated instance, while in the actual instance that will be realized, we shift the parameters one by one. We observe that Curate-Share is most sensitive to the *project arrival mix*, the *donor supply* and the *donation sizes*, and barely sensitive to the deadline: a 20% error in the arrival rates, alternating in sign across types, costs 8.5 points, a 20% shortfall in donor mass lowers the performance by 4.5, and making every gift one \$20 unit smaller, which lowers the average gift by 17.5%, costs 4.1, while a 20% shorter deadline costs 0.05 (Section EC.4.3, Table EC.3). Re-solving the plan on the true primitives recovers all of these losses, to within 0.1 points. The platform should therefore re-estimate arrival rates, donor supply and donation sizes often and re-solve its plan, while deadlines need little care.

6.6. Equity floors

We impose uniform floors $\beta \equiv \gamma \cdot \bar{\rho}^{\text{emp}}_{\text{unprot}}$ on all protected types and sweep the floor level γ (Figure 6), where $\bar{\rho}^{\text{emp}}_{\text{unprot}} = 0.686$ is the unweighted mean of the empirical funded rates of the 21 unprotected types, from the outcome labels (Table EC.1). Anchoring the floor to the unprotected rate lets γ read as parity: at $\gamma = 1$, every protected type is asked to be funded at least at the unprotected 2013 rate. Parity is not a stretch for the group, since in the 2013 data the protected set already does better than the unprotected one ($\hat{\rho}_{\text{prot}} = 0.725$ vs 0.686); what binds is that the floor is uniform *per type*, so the weakest protected type must reach the unprotected rate.

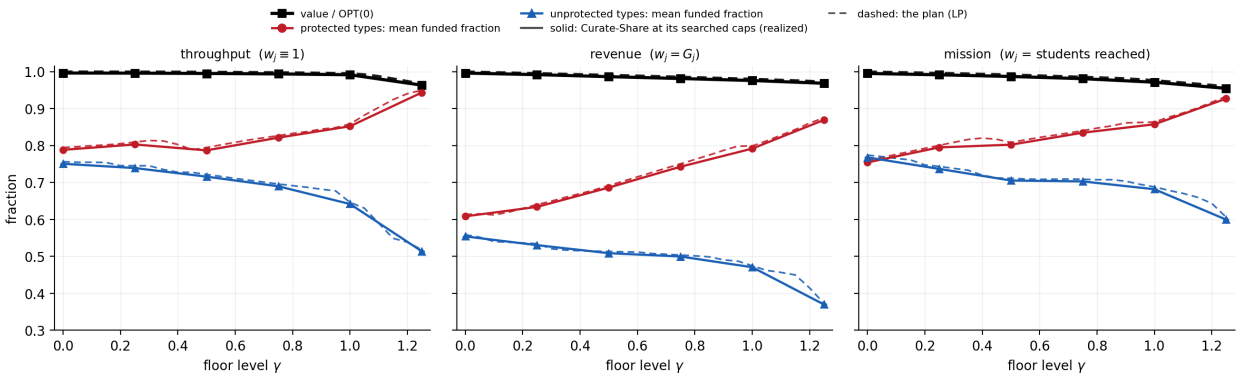


Figure 6 Bi-criteria guarantee for Curate-Share, by weighting. Panels: throughput ($w \equiv 1$), revenue ($w_j = G_j$) and mission (w_j = students reached). Solid lines are Curate-Share at its searched caps at $\gamma = 0, 0.25, \ldots, 1.25$; dashed lines are the LP, on a 0.05 grid of γ. Black squares: value over the unfloored optimum $\text{OPT}^{\text{unc}}_{\text{LP}}$ (dashed: $\text{OPT}_F/\text{OPT}^{\text{unc}}_{\text{LP}}$). Red circles and blue triangles: the mean funded fraction of the 21 protected and of the 21 unprotected types, unweighted over types, with abandoned types at zero.

The price of fairness, $1 - \mathrm{OPT}_F/\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$, is small throughout the range a platform would consider. It is continuous, convex and non-decreasing in γ, since the optimum of an LP is a concave function of the floor levels on its right-hand side. Under the mission weighting it is 1.84% at $\gamma = 0.9$, 2.35% at full parity, and 3.6% at $\gamma = 1.2$, a fifth above the unprotected 2013 rate, and only near the largest feasible floor, $\gamma^\star = 1.459$, does it climb steeply, to 6.5% at $\gamma = 1.4$ and 7.7% at $\gamma = 1.45$.

What this small price buys is service for the protected types the value-maximizing plan writes off. Under the mission weighting the unconstrained benchmark abandons nine types, five of them protected, and the protected group is then funded at a lower average rate than the unprotected one (0.759 against 0.774, unweighted over types with abandoned types at zero). Floors at $\gamma = 0.25$ already restore all five, raising the number of types the plan serves at a positive rate from 33 to 38, at a price of 0.37%; from $\gamma \geq 0.5$ the plan settles at 36, as two unprotected types leave it.

The online policy then delivers more protected service than the guarantee alone requires. We score Curate-Share and Curate-Dedicate on the same fair-LP optimum, each at its own searched caps. Without floors, the worst-off protected category is funded at rate zero under both, because no service rule can rescue a type the benchmark does not serve; the floors put the type back into the plan, and only then can sharing reach it. At the binding floor $\gamma = 0.9$, the worst-off protected category rises from 0.569 under Curate-Dedicate to 0.61 under Curate-Share, a gain of 7%, and the protected-group mean from 0.808 to 0.855. The competitive ratio at that floor is 0.950 under Curate-Dedicate against 0.995 under Curate-Share, and the floors lower Curate-Share's realized value by 1.9% of the unconstrained value, most of it (1.8 points) from the lower benchmark.

Additionally, when constraints are present, Curate-Share lifts every category compared to Curate-Dedicate, each at its own searched caps, and the protected and unprotected types gain about the same ($+5.8\%$ against $+6.2\%$, averaged over types). Capital, when it enters in Section 6.7, reinforces this, since about half of its value-optimal budget lands on protected types.

A protected set that donors actively *disfavor* is the sharper test. We run it in Section EC.4.2 on the ten types whose projects donors pick least often, i.e., with the fewest donations per live project-day. Without floors, the value-maximizing plan under the mission weighting drops four of them, and the floors restore them at a price of 0.26–3.03% of value as γ rises from 0.15 to parity. With the floors, Curate-Share funds each of them at 94% of its floor or more, and at parity serves the worst-off of them better than almost-funded-first, at 0.72 vs 0.59.

A platform can bring its highest-need categories to full parity with the rest, i.e., to the funded rate the unprotected types achieve in the 2013 data, for about 2.3% of total value under the mission weighting, and to a fifth above parity for 3.6% (Section EC.4.2).

6.7. Sponsor capital on the calibrated instance

In this subsection, we quantify the lever of Section EC.2 by solving the augmented program itself. For each budget B, we solve the enlarged program (EC.2) of Section EC.2.1 and report the true $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B)$, with budgets sized as a fraction of the full-funding cost $\sum_j \lambda_j G_j$.

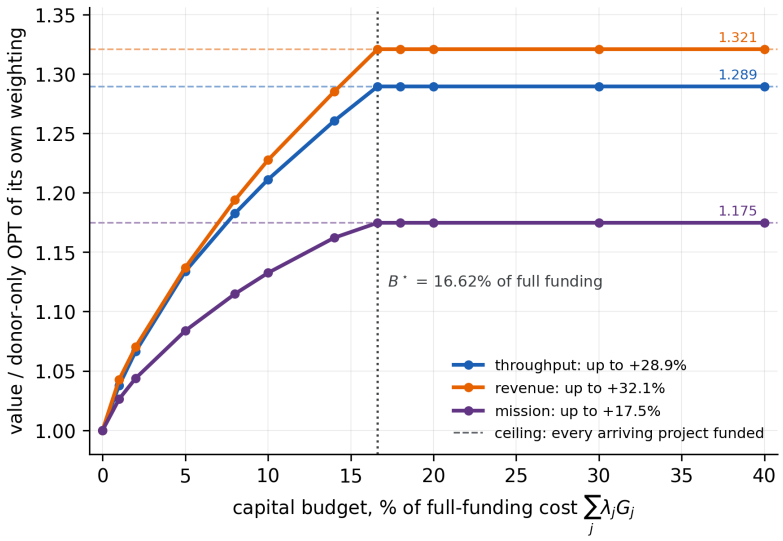


Figure 7 Capital on the calibrated instance, all three weightings. The augmented benchmark $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B)/\mathrm{OPT}_{\mathrm{LP}}$ against the budget, sized as a fraction of the full-funding cost $\sum_j \lambda_j G_j$, each weighting over its own donor-only optimum. The ceiling binds at the same budget, $B^\star = 16.62\%$ of full funding, under all three weightings.

The donor-only optimum is not the ceiling anymore. Capital buys back rationed projects, and the benchmark rises with the capital until the budget reaches $B^\star \approx \$30$k/day (about a sixth of the full funding cost). There, it funds *every* arriving project, and the ratio equals $\sum_j \lambda_j w_j/\mathrm{OPT}_{\mathrm{LP}}$ exactly (Figure 7), which is equal to 1.289, 1.321, 1.175, under throughput, revenue, and mission weighting, respectively.

When we run Curate-Share on the embedded chain against this richer benchmark, its realized value tracks the benchmark closely. At the canonical box, the competitive share under the throughput weighting, $\mathrm{CR}^{\mathrm{enl}}$, stays at about 0.98 as the budget grows; at its searched caps, it holds at 0.996 throughout. Capital enlarges the achievable value without enlarging donor capacity, so donor work per funded project falls from 6.32 matches at $B = 0$ to 4.96 at $B^\star$ under all three weightings.

We note that the return to budget is concave, i.e., the marginal value per unit of budget is highest at the start and falls as the budget grows, staying strictly positive up to $B^\star$ and vanishing only there. A platform with a small budget therefore captures more than its proportional share of the benefit. For instance, the first 5% of full funding, under a third of $B^\star$, already buys 43–48% of the potential gain across the three weightings.

Another interesting observation is that the capital goes mostly to finishing projects. At a budget of 10% of full funding, 83 to 89% of it, across the three weightings, is spent in the final quarter of a project's progress, the final stretch where individual gifts are smallest (Section EC.4.1); under the throughput weighting, the share is 97% at a budget of 5%.

Capital also makes fairness cheaper. Even when the unconstrained benchmark gets the same budget, the price of parity floors falls with the budget under every weighting and is exactly zero from $B^\star$ on, where the fair and the unconstrained programs both fund every project (Table EC.2); under the mission weighting, it is 2.35% with no capital, 0.45% at 5% of full funding and 0.10% at 14%.

6.8. Takeaways

The supporting analyses (the price of fairness, and robustness to perturbations and estimation error) are collected in Section EC.4, together with the calibration details. Two facts anchor the study. First, the benchmark is tight in practice: Curate-Share reaches 0.995–0.997 of $\mathrm{OPT}_{\mathrm{LP}}$, so the fluid frontier is attainable already at the 2013 scale, but not by a routing heuristic, which stops between 0.879 and 0.972 depending on the weighting. Second, equity is not a by-product of efficiency. Left to maximize value under the mission weighting, the optimum does not treat the protected types alike: it funds 16 of the 21 above their 2013 rate, most of them fully, and abandons the other five, along with four unprotected types; the floors cost about 2.3% of value for full parity, i.e., for funding every protected type at least at the mean funded rate of the unprotected types in the 2013 data, and the service-side levers, pooled exposure and a little capital, then deliver most to the categories that rationing and deadlines hurt most. A platform that pools its category exposure, chooses its assortments instead of showing everything, spends a small capital budget on its most rationed projects, sets explicit floors, and re-estimates arrivals, donor supply and donation sizes often will run close to the fluid frontier, with a guarantee and enforceable floors that no heuristic offers.

7. Concluding remarks

This paper asked how a donation-based crowdfunding platform should allocate scarce donor attention across concurrent all-or-nothing campaigns with goals and deadlines. Our answer has two stages. A fluid linear program thins the field before the system runs. Its value upper-bounds every policy, clairvoyant ones included, and its solution, the plan, fixes the donor attention each project type should receive at each funding level. Three modules then realize the plan online. Maintenance encodes the plan into per-project Markov chains and so keeps each type's load critical, Display splits the type's reserved rate evenly across its active projects through exact-rate assortment draws, while guaranteeing every admitted project a floor matching rate, and Admission control caps each type's active count. The policy is $1-\tilde{O}(1/\kappa)$-competitive against the fluid optimum (Theorem 3), and the rate is order-optimal up to polylogarithmic factors, since on a single-type deterministic-deadline instance every policy, online or offline, loses $\Omega(1/\kappa)$ (Proposition 2); the even split is what holds admission blocking to that order, because at critical load it makes each type's active count uniform over its admissible range, wherever the cap is set. When calibrated to the fairness-constrained benchmark, the same policy is simultaneously value-competitive and approximately fair at no additional online cost. In addition, the framework extends to handle sponsor capital as budgets under the control of the platform, via enriching the benchmark while preserving the guarantee.

On data calibrated to DonorsChoose, the policy comes within one percentage point of an attainable frontier and outperforms every greedy heuristic we test; it also carries the ex-ante guarantee and the enforceable floors those heuristics do not, and holds its performance under donor pool shrinkage while they lose theirs. Imposing fairness constraints that bring the highest-poverty classrooms to parity costs about 2.3% of value under the mission weighting, showing that achieving fairness does not have to come at the cost of efficiency. Capital

complements the donors: most of it goes to finishing projects at the stages that are less attractive to donors, where individual gifts are smallest. Its benefit is concave, so under a third of the budget that funds every project already buys almost half of the gain, and the same capital lowers the price of fairness, to zero once every project is funded. Re-solving the plan on the true primitives retains performance of the policy under every perturbation we test, while utilizing a stale plan based on biased estimates loses most when the arrival mix or the donor supply is the erroneous parameter.

References

Agrawal A, Catalini C, Goldfarb A (2015) Crowdfunding: Geography, social networks, and the timing of investment decisions. *Journal of Economics & Management Strategy* 24(2):253–274.

Alaei S (2014) Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM Journal on Computing* 43(2):930–972.

Alaei S, Malekian A, Mostagir M (2016) A dynamic model of crowdfunding. SSRN working paper 2737748, URL http://dx.doi.org/10.2139/ssrn.2737748.

Althoff T, Leskovec J (2015) Donor retention in online crowdfunding communities: A case study of DonorsChoose.org. *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*, 34–44 (ACM), URL http://dx.doi.org/10.1145/2736277.2741120.

Aminian MR, Ma W, Xin L (2026a) Online job selection: Reward rate vs. remaining value. *Management Science* Forthcoming.

Aminian MR, Manshadi V, Niazadeh R (2026b) Markovian search with ex ante constraints: Theory and applications to socially aware algorithmic hiring. *Management Science* URL http://dx.doi.org/10.1287/mnsc.2023.04079, articles in Advance.

Aminian MR, Niazadeh R, Nuti P (2026c) Optimal Bayesian online allocation of reusable resources. Available at SSRN 6384158.

Aminian MR, Niazadeh R, Nuti P (2026d) Stationary online contention resolution schemes. arXiv preprint arXiv:2603.21532.

Aouad A, Sarıtaç Ö (2022) Dynamic stochastic matching under limited time. *Operations Research* 70(4):2349–2383, URL http://dx.doi.org/10.1287/opre.2022.2293.

Aslan A, Bakir I, Çavdar B (2024) How does observational learning impact crowdfunding outcomes for backers, project creators and platforms? arXiv preprint arXiv:2401.01995.

Åstebro T, Fernández M, Lovo S, Vulkan N (2024) Herding in equity crowdfunding. *The RAND Journal of Economics* 55(3):403–441, URL http://dx.doi.org/10.1111/1756-2171.12474.

Aveklouris A, DeValve L, Stock M, Ward A (2025) Matching impatient and heterogeneous demand and supply. *Operations Research* 73(3):1637–1658, URL http://dx.doi.org/10.1287/opre.2022.0005.

Balseiro SR, Lu H, Mirrokni V (2025) Regularized online allocation problems: Fairness and beyond. *Manufacturing & Service Operations Management* 27(3):720–735, URL http://dx.doi.org/10.1287/msom.2022.0212.

Belavina E, Marinesi S, Tsoukalas G (2020) Rethinking crowdfunding platform design: Mechanisms to deter misconduct and improve efficiency. *Management Science* 66(11):4980–4997, URL http://dx.doi.org/10.1287/mnsc.2019.3482.

Belleflamme P, Lambert T, Schwienbacher A (2014) Crowdfunding: Tapping the right crowd. *Journal of Business Venturing* 29(5):585–609.

Bertsimas D, Farias VF, Trichakis N (2011) The price of fairness. *Operations Research* 59(1):17–31.

Burtch G, Ghose A, Wattal S (2013) An empirical examination of the antecedents and consequences of contribution patterns in crowd-funded markets. *Information Systems Research* 24(3):499–519.

Chaddad S, Hayel Y, Varma VS (2025) A stochastic model for optimizing crowdfunding campaign success. *Operations Research Letters* 61:107303, URL http://dx.doi.org/10.1016/j.orl.2025.107303.

Chakraborty S, Ma A, Swinney R (2023) Designing rewards-based crowdfunding campaigns for strategic (but distracted) contributors. *Naval Research Logistics* 70(1):3–20, URL http://dx.doi.org/10.1002/nav.22081.

Cong LW, Xiao Y (2024) Information cascades and threshold implementation: Theory and an application to crowdfunding. *The Journal of Finance* 79(1):579–629, URL http://dx.doi.org/10.1111/jofi.13294.

Deb J, Öry A, Williams KR (2024) Aiming for the goal: Contribution dynamics of crowdfunding. *American Economic Review* 114(12):3847–3876, URL http://dx.doi.org/10.1257/aer.20181851.

Désir A, Goyal V, Segev D, Ye C (2020) Constrained assortment optimization under the Markov chain-based choice model. *Management Science* 66(2):698–721.

DonorsChoose (2026a) Can I repost an expired or archived project? DonorsChoose Help Center, https://help.donorschoose.org/hc/en-us/articles/201937646-Can-I-repost-an-expired-or-archived-project, accessed September 27, 2026.

DonorsChoose (2026b) Impact. https://www.donorschoose.org/about/impact.html, accessed September 27, 2026.

DonorsChoose (2026c) What happens to donations when a project isn't funded? DonorsChoose Help Center, https://help.donorschoose.org/hc/en-us/articles/202000337-What-happens-to-donations-when-a-project-isn-t-funded, accessed September 27, 2026.

Du L, Hu M, Wu J (2022) Contingent stimulus in crowdfunding. *Production and Operations Management* 31(9):3543–3558, URL http://dx.doi.org/10.1111/poms.13782.

Feldman M, Svensson O, Zenklusen R (2021) Online contention resolution schemes with applications to Bayesian selection problems. *SIAM Journal on Computing* 50(2):255–300.

Feng Y, Niazadeh R, Saberi A (2024) Technical note—near-optimal Bayesian online assortment of reusable resources. *Operations Research* 72(5):1861–1873.

Gallego G, Ratliff R, Shebalov S (2015) A general attraction model and sales-based linear program for network revenue management under customer choice. *Operations Research* 63(1):212–232.

GoFundMe (2025) Common fundraising questions. https://www.gofundme.com/c/questions, accessed September 27, 2026.

GoFundMe (2026) Choosing your goal amount. GoFundMe Help Center, https://support.gofundme.com/hc/en-us/articles/4405145410331-Choosing-your-goal-amount, updated August 6, 2026; accessed September 27, 2026.

Golrezaei N, Nazerzadeh H, Rusmevichientong P (2014) Real-time optimization of personalized assortments. *Management Science* 60(6):1532–1551.

Grand View Research (2025) Crowdfunding market size, share & trends analysis report, 2025–2030. Report GVR-4-68040-006-1, https://www.grandviewresearch.com/industry-analysis/crowdfunding-market-report, page updated July 2026; accessed September 27, 2026.

Grötschel M, Lovász L, Schrijver A (1988) *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics* (Berlin, Heidelberg: Springer), URL http://dx.doi.org/10.1007/978-3-642-97881-4.

Gurvich I, Ward A (2014) On the dynamic control of matching queues. *Stochastic Systems* 4(2):479–523, URL http://dx.doi.org/10.1287/13-SSY097.

Hou T, He L, Keppo J, Guo X (2025) The role and optimization of staff picks in crowdfunding. SSRN working paper 5234468, URL http://dx.doi.org/10.2139/ssrn.5234468.

Huang A, Ivanov V (2025) Analysis of crowdfunding under the JOBS Act. Technical report, U.S. Securities and Exchange Commission, Division of Economic and Risk Analysis, https://www.sec.gov/files/dera-reg-cf-2505.pdf.

Indiegogo (2025) Indiegogo launches next-generation crowdfunding platform. Press release, PR Newswire, October 17, https://www.prnewswire.com/news-releases/indiegogo-launches-next-generation-crowdfunding-platform-302587678.html, accessed September 27, 2026.

Jackson CK, Mackevicius CL (2024) What impacts can we expect from school spending policy? Evidence from evaluations in the United States. *American Economic Journal: Applied Economics* 16(1):412–446, URL http://dx.doi.org/10.1257/app.20220279.

Jiang J, Ma W, Zhang J (2025) Tight guarantees for multiunit prophet inequalities and online stochastic knapsack. *Operations Research* 73(3):1703–1721, URL http://dx.doi.org/10.1287/opre.2022.0309.

KDD Cup (2014) KDD Cup 2014: Predicting excitement at DonorsChoose.org. Kaggle competition dataset, https://www.kaggle.com/c/kdd-cup-2014-predicting-excitement-at-donors-choose.

Kelly FP (1979) *Reversibility and Stochastic Networks* (Chichester: Wiley).

Kelly FP (1991) Loss networks. *The Annals of Applied Probability* 1(3):319–378.

Keppler SM, Li J, Wu DA (2022) Crowdfunding the front lines: An empirical study of teacher-driven school improvement. *Management Science* 68(12):8809–8828, URL http://dx.doi.org/10.1287/mnsc.2022.4331.

Keppler SM, Li J, Wu DA (2025) Stopping the revolving door: A causal and textual analysis of crowdfunding and teacher turnover. *Manufacturing & Service Operations Management* 27(6):1795–1813, URL http://dx.doi.org/10.1287/msom.2023.0206.

Kickstarter (2026) Creator handbook: Funding. https://www.kickstarter.com/help/handbook/funding, accessed September 27, 2026.

Kim C, Kannan PK, Trusov M, Ordanini A (2020) Modeling dynamics in crowdfunding. *Marketing Science* 39(2):339–365, URL http://dx.doi.org/10.1287/mksc.2019.1209.

Krengel U, Sucheston L (1977) Semiamarts and finite values. *Bulletin of the American Mathematical Society* 83(4):745–747.

Li Z, Duan JA, Ransbotham S (2020) Coordination and dynamic promotion strategies in crowdfunding with network externalities. *Production and Operations Management* 29(4):1032–1049, URL http://dx.doi.org/10.1111/poms.13150.

Liang A, Miao S, Xu Y, Zhou Q (2026) Right stage, right time: Dynamic seed-fund allocation with state-dependent donor response in healthcare crowdfunding. SSRN working paper 6843660, URL http://dx.doi.org/10.2139/ssrn.6843660.

Liu W, Ward AR, Zhang X (2026) Data-driven matching for impatient and heterogeneous demand and supply. *Mathematics of Operations Research* URL http://dx.doi.org/10.1287/moor.2025.1115.

Lucier B, Menache I, Naor J, Yaniv J (2013) Efficient online scheduling for deadline-sensitive jobs. *Proceedings of the 25th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '13)*, 305–314 (ACM), URL http://dx.doi.org/10.1145/2486159.2486187.

Manshadi V, Rodilitz S (2022) Online policies for efficient volunteer crowdsourcing. *Management Science* 68(9):6572–6590, URL http://dx.doi.org/10.1287/mnsc.2021.4220.

Manshadi V, Rodilitz S, Saban D, Suresh A (2025) Online algorithms for matching platforms with multichannel traffic. *Management Science* 71(9):7674–7691, URL http://dx.doi.org/10.1287/mnsc.2022.00910.

Mehrotra S (1992) On the implementation of a primal–dual interior point method. *SIAM Journal on Optimization* 2(4):575–601.

Mehta A (2013) Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science* 8(4):265–368.

Mollick E (2014) The dynamics of crowdfunding: An exploratory study. *Journal of Business Venturing* 29(1):1–16.

Morgan I (2022) Equal is not good enough: An analysis of school funding equity across the U.S. and within each state. Technical report, The Education Trust, https://edtrust.org/wp-content/uploads/2014/09/Equal-Is-Not-Good-Enough-December-2022.pdf.

National Center for Education Statistics (2026) Average amount of their own money that public and private school teachers spent on classroom supplies without reimbursement . . . : 2020–21. National Teacher and Principal Survey, table ntps2021_6801_t12ns, https://nces.ed.gov/surveys/ntps/estable/table/ntps/ntps2021_6801_t12ns, data from the 2020–21 survey, on spending during the 2019–20 school year; accessed October 7, 2026.

Samuel-Cahn E (1984) Comparison of threshold stop rules and maximum for independent nonnegative random variables. *The Annals of Probability* 12(4):1213–1216.

Song Y, Li Z, Sahoo N (2022) Matching returning donors to projects on philanthropic crowdfunding platforms. *Management Science* 68(1):355–375, URL http://dx.doi.org/10.1287/mnsc.2020.3930.

Strausz R (2017) A theory of crowdfunding: A mechanism design approach with demand uncertainty and moral hazard. *American Economic Review* 107(6):1430–1476, URL http://dx.doi.org/10.1257/aer.20151700.

Talluri K, van Ryzin G (2004) Revenue management under a general discrete choice model of consumer behavior. *Management Science* 50(1):15–33.

US Government Accountability Office (2020) K-12 education: School districts frequently identified multiple building systems needing updates or replacement. Report GAO-20-494, US Government Accountability Office, https://www.gao.gov/products/GAO-20-494.

Whittle P (1985) Partial balance and insensitivity. *Journal of Applied Probability* 22(1):168–176.

Whittle P (1988) Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability* 25A:287–298.

Wolff RW (1982) Poisson arrivals see time averages. *Operations Research* 30(2):223–231.

Wu J, Shi M, Hu M (2015) Threshold effects in online group buying. *Management Science* 61(9):2025–2040, URL http://dx.doi.org/10.1287/mnsc.2014.2015.

Yu Z, Xu Y, Tong L (2018) Deadline scheduling as restless bandits. *IEEE Transactions on Automatic Control* 63(8):2343–2358, URL http://dx.doi.org/10.1109/TAC.2018.2807924.

Zhang J, Savin S, Veeraraghavan SK (2023) Revenue management in crowdfunding. *Manufacturing & Service Operations Management* 25(1):200–217.

This page is intentionally blank. The e-companion (EC) starts next page.

EC.1. Further related work

Crowdfunding. A large empirical literature studies how these markets behave. Mollick (2014) traces the dynamics and success factors of all-or-nothing campaigns, Burtch et al. (2013) study how earlier contributions affect later contributors' decisions, Agrawal et al. (2015) show how geography and social ties shape the timing of investment, and Belleflamme et al. (2014) compare reward- and equity-based designs. Contributions also respond to a campaign's progress: they accelerate near the goal in the structural model of Kim et al. (2020), and sign-ups surge around group-buying thresholds (Wu et al., 2015). On DonorsChoose itself, Keppler et al. (2022, 2025) measure what funded projects do for students and teachers, and Althoff and Leskovec (2015) study which donors return. Dynamic models of one or two campaigns, each with a goal, random pledges and a deadline, study how backers react to one another: contributors who delay pledging until success looks likely (Chakraborty et al., 2023), herding under all-or-nothing and keep-it-all rules (Åstebro et al., 2024), information cascades under an all-or-nothing threshold (Cong and Xiao, 2024), and observational learning between two competing projects (Aslan et al., 2024). Others study a campaign's own levers: when to promote a campaign whose backing rises with its progress (Li et al., 2020), how a long-lived donor times contributions to coordinate arriving backers (Deb et al., 2024), and a creator's injection of friends-and-family money (Chaddad et al., 2025). The last is a single-campaign counterpart of our sponsor capital (Section EC.2), just as the seed budget of Liang et al. (2026) is a multi-campaign one. Mechanism design yields all-or-nothing targets as part of the optimal mechanism for a single campaign (Strausz, 2017), as well as platform rules that deter creator misconduct (Belavina et al., 2020). The models closest to ours, discussed in Section 1.1, study a single campaign, a single visibility lever, a seed budget over a fixed set of campaigns, or weekly recommendations to returning donors, whereas we decide admission and display in real time, subject to equity targets, with a guarantee against a fluid benchmark.

Online selection and assortment. Prophet inequalities (Krengel and Sucheston, 1977; Samuel-Cahn, 1984) and their modern descendant, online contention resolution schemes (OCRSs), measure the loss when stochastic demand meets a hard capacity. Alaei (2014) gives a single-resource guarantee that is tight for one unit, Feldman et al. (2021) the general framework, and Jiang et al. (2025) the tight multiunit guarantee and the knapsack extension. Mehta (2013) surveys the broader online matching literature. Choice-based assortment and revenue management have been studied in Talluri and van Ryzin (2004); Golrezaei et al. (2014); Désir et al. (2020). In network revenue management under discrete choice (Gallego et al., 2015), a fluid relaxation likewise guides an online rule. Our model differs in three respects: both projects and donors arrive online, values are all-or-nothing, and each project has a hard deadline that runs from its own arrival; none of these features is present in the classical formulations. The closest approach to ours is that of Feng et al. (2024), who study online assortment optimization of reusable resources via a simple random greedy OCRS that achieves near-optimal performance: an LP-guided online policy with discards, whose competitive ratio tends to one as capacities grow. Aminian et al. (2026c) sharpen the reusable-resource guarantee to the asymptotically optimal $1 - \sqrt{2/(\pi k)} + O(1/k)$, by introducing a stationary OCRS and discovering connections of OCRS with

Erlang-B loss systems. Aminian et al. (2026d) extend this stationarity idea to other feasibility environments for resource allocation. Our admission module is in this spirit, while our stochastic process is different: both sides (donors and projects) arrive stochastically, projects are stateful and need multiple donations, and the time a project needs until reaching its goal is random. We instead establish connections to the M/G/1/k processor-sharing queueing system, and utilize the admission probability under the stationary distribution. Nonprofit platforms also steer volunteers online: Manshadi and Rodilitz (2022) decide whom to notify for each arriving task, with an LP-based policy and a constant-factor guarantee, and Manshadi et al. (2025) decide which opportunity to recommend to each arriving volunteer when some traffic bypasses the recommendations. In all of these works, each match is valued on its own, whereas in our setting a project is worth something only when many donations reach its goal.

Scheduling and job selection. In online scheduling, jobs arrive over time, each with a value, a processing requirement and a deadline, and a job earns its value only if it completes by its deadline, a setting also studied under adversarial arrivals (Lucier et al., 2013). With random arrivals, Yu et al. (2018) schedule jobs with random sizes and hard deadlines on a limited number of servers, where unfinished work is penalized, and show that a Whittle index policy is asymptotically optimal in light traffic as the number of servers grows. A project in our problem resembles such a job, with the donations it needs as its processing; some of the key differences are that the service arrives as random donation amounts from stochastically arriving donors who choose among the projects shown, and the platform can steer this donor attention through assortments. Online job selection (Aminian et al., 2026a) allocates fixed capacity among adversarially arriving jobs with heterogeneous reward rates and service durations, where a preempted job yields a partial reward; a project in our setting earns nothing unless it is fully funded.

Loss networks and index policies. The count process stays tractable because of classical loss-network insensitivity (Kelly, 1979, 1991; Whittle, 1985) and PASTA (Wolff, 1982). The equity–efficiency trade-off we price connects to the price of fairness (Bertsimas et al., 2011) and to fairness regularizers in online allocation (Balseiro et al., 2025), whereas we impose floors on the funded rates of protected types. Deciding which types to keep active is also a dynamic resource-allocation problem, which places the paper near the index-policy literature: the admit-and-display choice echoes activity allocation in a changing environment (Whittle, 1988). Here, however, loss-network insensitivity collapses the control to a static, downward-closed region instead of a state-dependent index. What is new is coupling this machinery with a level-dependent payment process, donor attention routing via assortments, and an all-or-nothing goal plus hard deadline structure.

EC.2. Sponsor capital

Our core model in Section 2 treats donor attention as the platform's only resource. In the real world, however, donation-based crowdfunding platforms rarely run on traditional retail donors alone: the platforms routinely receive large donation amounts from governments, NGOs and non-profit foundations, and even from corporate philanthropy. Additionally, some of these platforms ask retail donors for an optional extra donation to the

platform itself (other than the fees, taxes, and operating costs of each project), which the platform may use to reach more projects and further its impact; e.g., DonorsChoose asks for an optional 15% of the donation amount. The platforms can then use these budgets to advance the projects they receive at different stages of their progress, including, but not limited to, topping up near-complete projects. The goal of this section is to incorporate this additional lever into the model and to show that the policy of Section 4 runs against the enlarged problem unchanged, at the same competitive factor, against a benchmark that is now better resourced and more powerful.

Let us first highlight some subtleties in how sponsor capital structurally compares with the regular donations we studied in Section 2. First, the platform has more control over how to disburse this money, as opposed to showing an assortment to the donors and observing what they select. Second, these amounts are typically larger and are used to partly fund various projects. Third, and most important, when a project fails, no money is actually disbursed: a donor's donation is refunded, and the opportunity to use it is lost, whereas sponsor capital can be reused by the platform to advance other projects.

Taking these aspects into account, we model sponsor capital as a budget that the platform can allocate arbitrarily over time. Since these budgets come and go dynamically and are of a larger scale, we impose an ex-ante limit on the average rate of budget consumption that the platform must respect at any point in time. We denote this limit by B; it represents the platform's expected rate of receiving capital, in a stationary (long-run) sense.

An important consequence of the third distinguishing factor above is that the "stake" in finishing a project depends on how much sponsor capital has already been spent on it; the funding level alone thus no longer determines a project's value to finish, and the state must also carry the sponsor capital spent so far on the project. Nevertheless, we show that one can still construct a fluid LP relaxation similar to that of Section 3, but on this enlarged state space, and again use it to guide an adapted version of our algorithmic framework of Section 4 that retains the same bi-criteria approximation factors.

With this in mind, the following definition formalizes what we mean by such an enlarged state.

DEFINITION EC.1 (ENLARGED STATE). The *enlarged state* of a project is the pair (g,k), where g is its funding level and k is the sponsor capital spent on it so far. A project is born at $(0,0)$. A donor match at (g,k) moves it to $(g+P,k)$ with $P\sim q_{j,g}$. A sponsor capital dedication of amount p moves it to $(g+p,k+p)$ and reduces the budget by p only if the project eventually succeeds. We refer to this latter jump in the state as the capital jump.

Note that the state space remains finite: each unit of capital advances the level by one unit and levels never decrease, so $0\le k\le g\le G_j$ and a type-j project has $O(G_j^2)$ enlarged states.

EC.2.1. The program

Let us denote by $M_j(g,k)$ the rate of donor matches at enlarged state (g,k), $N_j((g,k)\to h)$ the rate of capital jumps out of (g,k) to level h, and $\Pi_j(g,k)$ the rate of entries to (g,k). Let

$$\Phi_j(k) := \sum_g \bar{q}_{j,g}(G_j-g)\,M_j(g,k) + \sum_g N_j\big((g,\,k-(G_j-g))\to G_j\big) \tag{EC.1}$$

denote the success rate of type-j projects that use exactly k units of sponsor capital in total. We can now write our modified fluid LP:

$$\begin{aligned}
\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B) \;:=\; \max_{M,N\geq 0} \sum_j w_j S_j, \qquad & S_j = \sum_k \Phi_j(k), \qquad \Pi_j(0,0) = \lambda_j, \\
& \Pi_j(g,k) = \sum_{r<g} q_{j,r}(g-r)\, M_j(r,k) \;+\; \sum_{r<g} N_j\big((r,\, k-(g-r)) \to g\big), \\
& M_j(g,k) + \sum_{h>g} N_j\big((g,k)\to h\big) \;\leq\; \Pi_j(g,k), \qquad \Big(\sum_{g,k} M_j(g,k)\Big)_j \in \widehat{F}_+, \\
& \sum_j \sum_k k\,\Phi_j(k) \;\leq\; B, \qquad S_j \geq \lambda_j \beta_j \;\; (j\in\mathcal{P}).
\end{aligned} \tag{EC.2}$$

The new constraint is the budget constraint in the last line, which captures the platform's steady-state budget limit. The other constraints are analogous to their counterparts in Section 3.1, now including the effect of the new capital jump variables $N_j((g,k)\to h)$. Note that every constraint is still linear in our variables (M,N), including the budget constraint. In particular, including the capital spent in the state allows us to essentially condition on the success of the project, by accounting for the capital expenditure only at the moment of completion. Thus, we can capture the true expected rate of expenditure exactly, without multiplying the capital spent at each level by the success probability from that level, which is an endogenous quantity and would make the program nonlinear.

Two points are worth stating. First, capital never hurts: the platform can decide to ignore the budget and set all the new capital variables to zero, so $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B) \geq \mathrm{OPT}_{\mathrm{LP}}$ for every B, with equality at $B=0$. Second, the gain concerns total value, not any single type. The augmented optimum may reallocate donors across types, and making one type cheaper to complete can pull donor attention from other types toward it and lower the other types' success rates even as total value rises. All statements in this section are made against the augmented optimum itself.

Next, we prove that this enlarged program again serves as an upper bound for every feasible policy that also has access to the capital. Let us first formally define what budget compliance means for a policy.

DEFINITION EC.2 (BUDGET-FEASIBLE POLICY). For a policy π and a horizon T write

$$\Psi^{\pi,T} \;:=\; \frac{1}{T}\,\mathbb{E}_\pi\Big[\sum_{\text{projects } \ell \text{ funded by } T} k_\ell\Big],$$

where k_ℓ is the capital accumulated on project ℓ at the moment it is funded. The policy π is *budget-feasible at* B if $\limsup_{T\to\infty} \Psi^{\pi,T} \leq B$.

THEOREM EC.1 **(The enlarged program bounds every policy).** *Let π be any policy that is non-anticipating with respect to payment and choice realizations as well as budget-feasible at B in the sense of Definition EC.2. It may condition on the entire past of every project in the system, may randomize, and may know the future arrival stream. Then $J(\pi) \leq \mathrm{OPT}^{\mathrm{enl,unc}}(B)$, the optimum of (EC.2) with the fairness constraints dropped. If in addition π is fair in the sense of Section 2.3, then $J(\pi) \leq \mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B)$.*

EC.2.2. Running the plan

Similar to Section 4.1, we use (EC.2) to guide our algorithm of Section 4, adapted to the enlarged state space.

First, let (M, N) be feasible for (EC.2) and set

$$c_j(g,k) := \frac{M_j(g,k)}{\Pi_j(g,k)}, \qquad \sigma_j\big((g,k)\to h\big) := \frac{N_j\big((g,k)\to h\big)}{\Pi_j(g,k)}.$$

As before, one can now run the per-project chain on the phases $\{(g,k)\}\cup\{G_j,\star\}$ as follows: upon entering (g,k), the project either stays active at that level, awaiting a donor match, with probability $c_j(g,k)$; uses capital to raise its level to h with probability $\sigma_j((g,k)\to h)$ (for each feasible h); or is otherwise absorbed at the null state $\star$. Note that if, upon entering a state, the realized decision is to jump to a higher level using capital, the jump happens immediately, and we again choose among the three options above (waiting for a donor, using further capital, or absorbing at $\star$) upon entering the new state. See Section EC.3.1 for further details on how to run this chain, which includes collapsing it first to a donor-pure chain. Additionally, denote by $\hat{b}_j(g,k)$ and $\hat{a}_j(g,k)$ its entry and active (waiting for a donor) probabilities at any state (g,k) as before. Then, the following lemma holds:

LEMMA EC.1 **(The enlarged chain matches the plan).** *Let (M,N) be feasible for (EC.2) and run the per-project chain of Section EC.2.2 on the enlarged phases. Then we have $\Pi_j(g,k) = \lambda_j\hat{b}_j(g,k)$ and $M_j(g,k) = \lambda_j\hat{a}_j(g,k)$, the chain is absorbed at G_j with probability $\rho_j^{\mathrm{enl}} := S_j/\lambda_j$, and $\lambda_j\mathbb{P}[\text{funded with } k] = \Phi_j(k)$.*

Write $\bar{K}_j^{\mathrm{don,enl}} := \sum_{(g,k)}\hat{a}_j(g,k)$ for the mean number of donor-consuming ticks per arrival (unconditional on the outcome and including zero-donation arrivals, exactly as in Section 4.1). As before, we can write the aggregate donor-attention load offered by all type-j projects as

$$\bar{y}_j^{\mathrm{enl}} \;=\; \lambda_j\,\bar{K}_j^{\mathrm{don,enl}} \;=\; \sum_{g,k} M_j(g,k)\,. \tag{EC.3}$$

As in Lemma 4, the fluid LP ensures that $(\bar{y}_j^{\mathrm{enl}})_j \in \widehat{F}_+$, so by downward closure, every rate vector obtained by setting some coordinates of $\bar{y}^{\mathrm{enl}}$ to 0 is feasible and implementable. Also note that $\bar{K}_j^{\mathrm{don,enl}} \le G_j$ deterministically, since each donor-consuming visit strictly increases the level while k increments only at capital jumps, which consume no donor.

The display rule and the admission rule are unchanged, except that a project faces the admission rule only after its capital jumps on arrival. Note that the extension changes the resulting Markov chain, as discussed in Section EC.3.1, but the deadline analysis of Section 4.2 applies unaltered; we write $\delta_j^{\mathrm{enl}} := \mathbb{P}[\tau_j(u_j) > D_j]$ for the resulting deadline-miss probability of an admitted project on the enlarged chain, on which capital jumps take no time. It remains to show that the count process induced by these rules still has the uniform product-form distribution that our admission bound needs.

LEMMA EC.2 **(The enlarged count is a balanced-sharing network).** *Run our policy of Section 4.4 on the enlarged phases at reserved rate $\bar{y}_j^{\mathrm{enl}}$, admitting to a finite downward-closed region. Then the configuration process has the insensitive product form of Theorem 1, and the count distribution depends on the funding dynamics only through the mean $\bar{K}_j^{\mathrm{don,enl}}$; under critical load — which holds identically, since $\bar{y}_j^{\mathrm{enl}} = \lambda_j \bar{K}_j^{\mathrm{don,enl}}$ by (EC.3) — the count is uniform on the region; and on the box $\mathcal{K}^{\square}$ derived from the enlarged program the admission blocking is $1/(K_j^{\square}+1)$ with $K_j^{\square} = \lfloor \bar{y}_j^{\mathrm{enl}}/u_j \rfloor$.*

Before stating the main theorem of this section, we assume that the augmented optimum satisfies the analogue of the served-set convention of Remark 5 ($\rho_j^{\mathrm{enl}} > 0$ for every type), noting that some projects may be absorbed immediately upon arrival, now possibly at the success level G_j using capital alone, in which case they never wait for a donor and their reward is earned directly.

THEOREM EC.2 **(Competitive ratio with capital).** *Run the policy of Section EC.2.2 based on an optimum of (EC.2). Then*

$$J(\pi^{\mathrm{enl}}) \;\geq\; \Big(\min_{j:\, \bar{K}_j^{\mathrm{don,enl}} > 0} (1-\alpha_j^{\mathrm{block}}) \big(1 - \delta_j^{\mathrm{enl}}/\rho_j^{\mathrm{enl}}\big)_+ \Big)\, \mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B), \qquad \alpha_j^{\mathrm{block}} = \tfrac{1}{K_j^{\square}+1}, \tag{EC.4}$$

and under the clock scaling of Theorem 3, with the budget B also scaled alongside the market as the new third side of the platform, both losses are $\tilde{O}(1/\kappa)$, so $J(\pi^{\mathrm{enl}}) \geq (1-\tilde{O}(1/\kappa))\,\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B) \geq (1-\tilde{O}(1/\kappa))\,\mathrm{OPT}_{\mathrm{LP}}$.

Definition EC.2 constrains only the long-run rate, so that Theorem EC.1 bounds every policy meeting it. A policy can meet a stronger requirement by respecting the budget at any point in time, and ours does: since it starts in stationarity (Section 4.4), its expected spending never exceeds the plan's at any point in time, so it is budget-feasible in that stronger sense, and not only in the limit.

As a concluding remark for this section, we highlight that capital and donor attention are substitutes for advancing a project, but they help in different places. Curate-Share already extracts all but a $\tilde{O}(1/\kappa)$ fraction of the value donor attention can support, so the role of capital is not to fix the admission side but to enlarge what is achievable: it funds the projects donors cannot reach, and the same policy captures its usual share of the larger optimum.

EC.3. Deferred proofs

This appendix collects the proofs deferred from Section 2, Section 3 and Section 4, in the order in which the statements appear.

Proof of Lemma 1. Fix a donor type i, a point $y_i \in \widehat{F}_i$ and a target $0 \leq y_i' \leq y_i$ coordinatewise. Write $\varphi_i(R) := (\varphi_i(j \mid R))_{j\in\mathcal{J}}$ for the selection vector of an assortment R, and fix a representation $y_i = \sum_k \alpha_k \varphi_i(R_k)$ with $\alpha_k \geq 0$ and $\sum_k \alpha_k = 1$, which exists because $\widehat{F}_i$ is the convex hull of the finitely many such vectors. Set $\theta_j := (y_i')_j/(y_i)_j$ where $(y_i)_j > 0$ and $\theta_j := 0$ where $(y_i)_j = 0$; this is well defined, and $\theta_j \in [0,1]$, because $0 \leq (y_i')_j \leq (y_i)_j$ forces $(y_i')_j = 0$ whenever $(y_i)_j = 0$. For each k the scaled vector $z^{(k)} := (\theta_j\, \varphi_i(R_k)_j)_j$ satisfies $z^{(k)} \leq \varphi_i(R_k)$ coordinatewise, so under weak substitutability sub-assortment sampling (Feng et al.,

2024) realizes $z^{(k)}$ exactly by a distribution over subsets of R_k, and in particular $z^{(k)} \in \widehat{F}_i$. Recombining with the same weights gives $\sum_k \alpha_k z_j^{(k)} = \theta_j\,(y_i)_j = (y_i')_j$ for every j, both sides vanishing when $(y_i)_j = 0$, so $y_i' = \sum_k \alpha_k z^{(k)}$ lies in $\widehat{F}_i$ by convexity. □

Proof of Proposition 1. Fix a policy and a horizon T, and let $M_j^{\pi,T}(g) := \frac{1}{T}\,\mathbb{E}_\pi[\#\{\text{type-}j \text{ projects matched at level } g \text{ by time } T\}]$ be its empirical match rate; define $\Pi_j^{\pi,T}$ from $M_j^{\pi,T}$ by (3). We check that $(M_j^{\pi,T})$ is feasible for (P) at every finite T, exactly.

Reach recursion and (Project Availability). Because payments are strictly positive, a type-j project reaches each transient level $g \geq 1$ at most once, at the instant a match at some level $r < g$ draws the payment $g - r$. Payments are drawn i.i.d. from $q_{j,r}$ at the match epoch, independently of every decision made so far (the draw is revealed only then; payment homogeneity, Assumption 1(c)), so in long-run expectation the rate of reaching g is $\sum_{r<g} q_{j,r}(g-r)\,M_j^{\pi,T}(r) = \Pi_j^{\pi,T}(g)$ exactly: whether the kth type-j match occurs by T is determined before that match's payment is revealed, so Wald's identity applies with no boundary term. Concretely, let ξ_k be the payment drawn at the kth type-j match at level r: whether that match occurs by T is decided before ξ_k is drawn, and ξ_k is independent of everything before it, so by Wald's identity the expected number of level-r matches by T that draw $g - r$ is $q_{j,r}(g-r)$ times the expected number of level-r matches by T, that is, $q_{j,r}(g-r)\,T M_j^{\pi,T}(r)$; and there is no boundary term because a match and its payment happen at the same instant, so no match counted by T is still waiting for its payment. A project must reach g before it can be matched there and is matched at g at most once, so $M_j^{\pi,T}(g) \leq \Pi_j^{\pi,T}(g)$; at $g = 0$ every reach is a distinct admitted arrival, so the realized rate of type-j arrivals admitted is at most λ_j, and $\Pi_j(0) := \lambda_j$ by definition, whence $M_j^{\pi,T}(0) \leq \Pi_j(0)$.

(Donor Availability). Each completed match (i, j) is preceded by exactly one selection of type j by a type-i donor. Writing $a_{i,R}^{\pi,T}$ for the rate at which type-i donors are shown assortment R, choice depends on the assortment only through the project types it contains (Assumption 1(a)), so the type-i selection vector $\sum_R a_{i,R}^{\pi,T}(\varphi_i(j \mid R))_j$ lies in $\mu_i \widehat{F}_i$ as a convex combination of choice vertices — for *any* policy, since even a clairvoyant one cannot foresee a donor's realized choice. As matches cannot exceed selections, summing over i and using downward closure of $\widehat{F}_+$ (Assumption 1(b)) gives $\left(\sum_g M_j^{\pi,T}(g)\right)_j \in \widehat{F}_+$.

Objective. Each type-j success is a match at some level g whose payment overshoots G_j, an event of probability $\bar{q}_{j,g}(G_j - g)$ drawn at the match epoch, so the value accrued by T is $\sum_j w_j \sum_g \bar{q}_{j,g}(G_j - g)\,M_j^{\pi,T}(g) = \sum_j w_j S_j^{\pi,T}$ by (2) and the same Wald argument.

Thus $(M_j^{\pi,T})$ is feasible for (P), so the value earned by T satisfies $\frac{1}{T}\mathbb{E}_\pi[\text{value earned by } T] = \sum_j w_j \sum_g \bar{q}_{j,g}(G_j - g)\,M_j^{\pi,T}(g) \leq \mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$ at every T, since every success is counted at its match epoch. Since (1) counts value when it is earned, taking $\limsup_{T\to\infty}$ gives $J(\pi) \leq \mathrm{OPT}_{\mathrm{LP}}^{\mathrm{unc}}$ directly.

For the fair bound, along a subsequence realizing $J(\pi)$, the empirical match rates $M_j^{\pi,T_n}(g)$ converge (Bolzano–Weierstrass, the rates being finitely many bounded scalars) to a limit feasible for (P); for a fair policy $\liminf_T S_j^{\pi,T}/\lambda_j \geq \beta_j$ passes to the limit, giving $S_j^\pi \geq \lambda_j \beta_j$, so the limit is also feasible for (P_F). Hence $\mathrm{OPT}_F \geq \sum_j w_j S_j^\pi = J(\pi)$. □

Proof of Lemma 2. Upward-monotonicity of the chain ($p \geq 1$ on every transient transition) implies that each entry into a transient level g is associated with a unique preceding active level $r < g$ from which the survival transition led to g. The probability of entering g is therefore obtained by conditioning on this preceding active level: the chain is active at r with probability $\hat{a}_j(r)$, and from active r the survival kernel (13) produces destination g with probability $q_{j,r}(g-r) \cdot c_j(g)$. Aggregating only the entry contribution (without the continuation probability $c_j(g)$, which is applied at the destination): $\hat{b}_j(g) = \sum_{r<g} \hat{a}_j(r) q_{j,r}(g-r)$. The active probability is then $\hat{a}_j(g) = c_j(g)\hat{b}_j(g)$.

The absorbing identity (17) comes from $P_j(r, G_j) = \bar{q}_{j,r}(G_j - r)$ in (13): a transition from active r to G_j uses no further continuation probability, so the contribution is $\hat{a}_j(r)\bar{q}_{j,r}(G_j - r)$, and summing over r gives the total absorption probability at G_j. □

Proof of Lemma 3. We first identify the chain quantities $(\hat{b}_j, \hat{a}_j)$ of Lemma 2 with the LP quantities. The LP recurrence for the cumulative match flow gives $\Pi_j(g)/\lambda_j$ (the entry probability to level g) and $M_j(g)/\lambda_j$ (the active probability at level g); these satisfy the same lower-triangular recurrence (16) with kernel $q_{j,r}(g-r)$ and boundary condition $\hat{b}_j(0) = 1$. Since the system (16) is uniquely solvable, the chain's solution coincides with the LP quantities of (10):

$$\hat{b}_j(g) = b_j(g) = \Pi_j(g)/\lambda_j, \qquad \hat{a}_j(g) = a_j(g) = M_j(g)/\lambda_j,$$

consistent with (10); in view of this identification we drop the hats from here on, confirming $c_j(g) = a_j(g)/b_j(g) = M_j(g)/\Pi_j(g)$. Substituting into (17) of Lemma 2:

$$s_j \;=\; \sum_{r=0}^{G_j-1} \frac{M_j(r)}{\lambda_j}\, \bar{q}_{j,r}(G_j - r) \;=\; \frac{1}{\lambda_j} \sum_{r=0}^{G_j-1} \bar{q}_{j,r}(G_j - r)\, M_j(r) \;=\; \frac{S_j^*}{\lambda_j} \;=\; \rho_j,$$

by the definition of S_j in (4) and ρ_j in (12). For the donor budget, taking expectations in (15) gives $\bar{K}_j^{\mathrm{don}} = \sum_g \hat{a}_j(g)$, and the same identification yields (18). □

Proof of Lemma 4. Step 1: every $y \in \widehat{F}_+$ is implementable. By definition of the Minkowski sum $\widehat{F}_+ = \sum_i \mu_i \widehat{F}_i$, there exist vectors $y_i \in \widehat{F}_i$ such that

$$y \;=\; \sum_i \mu_i\, y_i.$$

Since each $\widehat{F}_i$ is the convex hull of the assortment-choice vectors $\{(\varphi_i(j \mid R))_j\}_R$, each y_i can be written as

$$y_i \;=\; \sum_R \mathcal{D}_i(R)\, (\varphi_i(j \mid R))_j$$

for some $\mathcal{D}_i \in \Delta(2^{\mathcal{J}})$. Substituting coordinate-wise, $y_j = \sum_i \mu_i \sum_R \mathcal{D}_i(R)\, \varphi_i(j \mid R) = \sum_i \mu_i\, \mathbb{E}_{R\sim\mathcal{D}_i}[\varphi_i(j \mid R)]$, which is the LHS of (19). When $(y_i)_j = 0$ for some i, j, then $\mathcal{D}_i$ puts mass only on assortments R with $\varphi_i(j \mid R) = 0$ (any other vertex would contribute a strictly positive term to $(y_i)_j$), so under $\mathcal{D}_i$ no displayed assortment contains j with positive selection probability. Finally, only y is needed, not its blocks $(y_i)_i$: since

$\widehat{F}_+ = \sum_i \mu_i \widehat{F}_i$ is a Minkowski sum, a linear optimization over $\widehat{F}_+$ is I separate linear optimizations, one over each $\widehat{F}_i$, so every vertex it returns comes already decomposed by donor type; the exact Carathéodory scheme of Aminian et al. (2026b) run with this oracle therefore returns, in polynomial time, polynomially many such donor-decomposed vertices and weights realizing y, and $\mathcal{D}_i$ puts the kth weight on the type-i assortment of the kth vertex.

Step 2: the certificate. By (15) and $a_j(g) = M_j(g)/\lambda_j$ we have $\lambda_j \bar{K}_j^{\mathrm{don}} = \sum_g M_j(g)$, and constraint (Donor Availability), (6), places the total per-type match-rate vector in the donor polytope, so $\bar{\mathbf{y}} = (\lambda_j \bar{K}_j^{\mathrm{don}})_j = \big(\sum_g M_j(g)\big)_j \in \widehat{F}_+$.

Step 3: downward closure. By Assumption 1(b), each $\widehat{F}_i$ is downward closed in $\mathbb{R}_+^J$, and the Minkowski sum $\widehat{F}_+ = \sum_i \mu_i \widehat{F}_i$ of downward-closed sets in the non-negative orthant is downward closed. Hence every $y \le \bar{\mathbf{y}}$ lies in $\widehat{F}_+$; in particular so does $y^* := (\bar{y}_j \mathbf{1}\{m_j \ge 1\})_j$, at every count m. Step 1 applies to each such y, which is the statement. □

Proof of Lemma 5. The pathwise bound $\delta_j \le \mathbb{P}[\Xi_j > u_j D_j]$ was discussed in the text. For the closed form, the initial coin sits in the initial distribution, not in $P^{(j)}$: a coin-failed arrival (probability $1 - c_j(0)$) absorbs at once with $K_j^{\mathrm{don}} = 0$, while $e_0^\top (P^{(j)})^k \mathbf{1}$ is the k-tick survival probability conditional on an active start. The unconditional survival probabilities therefore carry the coin, which is the $\sigma_k^{(j)}$ of Lemma 5,

$$\sigma_k^{(j)} = \mathbb{P}[K_j^{\mathrm{don}} > k] = c_j(0)\, e_0^\top \big(P^{(j)}\big)^k \mathbf{1}; \tag{EC.5}$$

payments are positive, so $P^{(j)}$ is strictly upper triangular and $(P^{(j)})^{G_j} = 0$, which is why the sum terminates at $k = G_j - 1$. Also $\mathbb{E}K_j^{\mathrm{don}} = \sum_{k\ge 0} \sigma_k^{(j)} = \bar{K}_j^{\mathrm{don}}$. Since the timing of the ticks (donor matches) is independent of the realized trajectory of the chain, and thus of the number of ticks K_j^{don} until absorption, conditional on K_j^{don}, Ξ_j would be a sum of K_j^{don} unit-rate exponentials, which is $\mathrm{Erlang}(K_j^{\mathrm{don}}, 1)$. Therefore, Ξ_j would be the mixture of Erlangs (21). For its tail we use the duality of Erlang and Poisson: $\mathrm{Erlang}(\ell, 1)$ exceeds x if and only if a rate-1 Poisson process has fewer than ℓ arrivals by time x, so $\mathbb{P}[\mathrm{Erlang}(\ell,1) > x] = \mathbb{P}[f < \ell]$ for $f \sim \mathrm{Poisson}(x)$. Taking f independent of K_j^{don} and mixing over ℓ gives $\mathbb{P}[\Xi_j > x] = \mathbb{P}[f < K_j^{\mathrm{don}}]$, a convex combination of the left tails of the $\mathrm{Poisson}(x)$ pmf, cut at K_j^{don}. Exchanging the two sums, with $\mathbb{P}[f = k] = e^{-x}x^k/k!$ and $\mathbb{P}[K_j^{\mathrm{don}} > k] = \sigma_k^{(j)}$ by (EC.5),

$$\mathbb{P}[\Xi_j > x] = e^{-x} \sum_{k=0}^{G_j - 1} \sigma_k^{(j)} \frac{x^k}{k!} \;\le\; e^{-x} \sum_{k=0}^{G_j - 1} \frac{x^k}{k!} = \mathbb{P}\big[\mathrm{Poisson}(x) < G_j\big]. \tag{EC.6}$$

The equality is the exact phase-type tail (a rate-1 Poisson process delivers k ticks with probability $e^{-x}x^k/k!$, and $\Xi_j > x$ iff fewer than K_j^{don} ticks arrive by effort x); the inequality is the skip-free worst case $\sigma_k^{(j)} \equiv 1$, i.e. $\mathrm{Erlang}(G_j, 1)$. Combining (20)–(EC.6),

$$\delta_j \;\le\; \mathbb{P}\big[\mathrm{Poisson}(u_j D_j) < G_j\big] = e^{-u_j D_j} \sum_{k=0}^{G_j - 1} \frac{(u_j D_j)^k}{k!} = O\big((u_j D_j)^{G_j - 1} e^{-u_j D_j}\big), \tag{EC.7}$$

the $O(\cdot)$ holding as $u_j D_j \to \infty$ at fixed G_j: for $u_j D_j \geq G_j$ the terms $(u_j D_j)^k/k!$ increase in k, so the sum is at most G_j times its last term. □

Proof of Corollary 1. Let $A = \{\text{absorb at } G_j\}$ and $B = \{\tau_j(u_j) > D_j\}$. Then $\mathbb{P}[A] = \rho_j$ by Lemma 3 and $\mathbb{P}[B] = \delta_j$ by definition. Hence $\mathbb{P}[A \cap B^c] = \mathbb{P}[A] - \mathbb{P}[A \cap B] \geq \mathbb{P}[A] - \mathbb{P}[B] = \rho_j - \delta_j$. Since a probability is always ≥ 0, we obtain $\mathbb{P}[A \cap B^c] \geq (\rho_j - \delta_j)_+$. □

Proof of Theorem 1. The virtual configuration process has finitely many states, since $\mathcal{F}$ is finite and each active label sits at one of finitely many levels. It is irreducible: from any configuration it can reach the empty one, since every active label is absorbed within G_j donor ticks, and from the empty one it can reach any configuration whose count lies in $\mathcal{F}$ and whose labels sit at levels the per-label chain can visit, admitting one project at a time, which $\mathcal{F}$ allows because it is downward closed. Being finite and irreducible, it is positive recurrent with a unique stationary distribution (in continuous time there is no periodicity to rule out), so it suffices to exhibit one distribution satisfying the balance equations. First, note that transitions in the Markov chain are either due to some type-j project arrival or some type-i donor arrival. However, because the donors will be routed independently, we get a Poisson decomposition of our donor events based on which type-j project they get matched to. Hence, we can partition the CTMC's transition events based on the project type they correspond to. In the following, we then verify partial balance for each of these types: for each j, the flow into x along type-j transitions matches the flow out by type-j transitions. Out of x, type-j fires service ticks (donor arrival event) at total rate $\frac{\bar{y}_j}{n_j}\sum_g x_{j,g} = \bar{y}_j$ and occupies a new slot (project arrival event) at rate $\lambda_j a_j(0)\,\mathbf{1}\{n + e_j \in \mathcal{F}\}$ (each arrival enters the active pool with probability $a_j(0) = c_j(0)$, and otherwise changes nothing); we match the two separately. Both matchings use the visit probabilities of Lemma 2 through

$$\sum_g a_j(g)\, P^{(j)}_{gh} \;=\; a_j(h)\,\mathbf{1}\{h > 0\}, \qquad a^{(j)} q^{(j)} \;=\; a_j(0), \tag{EC.8}$$

the first being (16) written through the kernel (13), and the second following by summing it over h with $\sum_h P^{(j)}_{gh} = 1 - q^{(j)}_g$, where $q^{(j)}_g$ is the absorption probability of Section 4.5.

(i) Service in + arrival in = service out. Project arrival inflow to x comes from $x - e_{j,0}$ and a non-absorbing donor event initiates from $x + e_{j,g} - e_{j,h}$ (a $g \to h$ advance). From (24), $\dfrac{\pi(x - e_{j,0})}{\pi(x)}\,\lambda_j a_j(0) = \dfrac{\bar{y}_j}{n_j}\,x_{j,0}$ and $\dfrac{\pi(x + e_{j,g} - e_{j,h})}{\pi(x)} = \dfrac{x_{j,h}}{x_{j,g}+1}\dfrac{a_j(g)}{a_j(h)}$, so the sum of these two inflows divided by $\pi(x)$ is

$$\frac{\bar{y}_j}{n_j}\Big[x_{j,0} + \sum_h \frac{x_{j,h}}{a_j(h)} \sum_g a_j(g)\, P^{(j)}_{gh}\Big] = \frac{\bar{y}_j}{n_j}\Big[x_{j,0} + \sum_{h>0} x_{j,h}\Big] = \frac{\bar{y}_j}{n_j}\sum_h x_{j,h} = \bar{y}_j,$$

the middle step using (EC.8), whose left side vanishes at $h = 0$ and equals $a_j(h)$ for $h > 0$.

(ii) Completion in = arrival out = $\lambda_j a_j(0)\,\mathbf{1}\{n + e_j \in \mathcal{F}\}$. When $n + e_j \in \mathcal{F}$, completion inflow comes from $x + e_{j,g}$ (a level-g project absorbs):

$$\sum_g \frac{\pi(x + e_{j,g})}{\pi(x)}\,\frac{\bar{y}_j}{n_j + 1}(x_{j,g}+1)\, q^{(j)}_g = \varrho_j\,\frac{\bar{y}_j}{\bar{K}^{\mathrm{don}}_j}\,a^{(j)} q^{(j)} = \lambda_j a_j(0),$$

since $\varrho_j \bar{y}_j / \bar{K}_j^{\mathrm{don}} = \lambda_j$ and $a^{(j)} q^{(j)} = a_j(0)$ by (EC.8); when $n + e_j \notin \mathcal{F}$ neither term exists. Summing (i)–(ii) over j gives global balance. Summing (24) over phases at fixed counts (multinomial theorem) yields (25) and the conditional multinomial. □

Proof of Theorem 2. Since the chain starts in stationarity, its distribution at every $t \geq 0$ is π_{joint}, so by PASTA a type-j arrival sees the stationary count and is admitted with probability $1 - \alpha_j^{\mathrm{block}}$; conditional on admission it is funded by its deadline with probability $\geq (\rho_j - \delta_j)_+$ (Corollary 1), so it is successfully funded with probability $\geq (1 - \alpha_j^{\mathrm{block}})\,(\rho_j - \delta_j)_+$. By stationarity this probability, ρ_j^{ALG}, does not depend on the arrival time, which is (30), and type-j projects are therefore funded at the long-run rate $\lambda_j \rho_j^{\mathrm{ALG}}$. □

Proof of Corollary 2. Multiplying (30) by $\lambda_j w_j \geq 0$ and summing over j gives (31), since $J(\pi) = \limsup_{T\to\infty} \sum_j w_j S_j^{\pi,T} = \sum_j \lambda_j w_j \rho_j^{\mathrm{ALG}}$ by (1) and Theorem 2. For (32), write $(\rho_j - \delta_j)_+ = \rho_j (1 - \delta_j/\rho_j)_+$ (valid as $\rho_j > 0$, by the convention of Section 3.2) and factor $\min_j$ out of $\sum_j (1 - \alpha_j^{\mathrm{block}})(1 - \delta_j/\rho_j)_+ \cdot \lambda_j w_j \rho_j$, and note that $\mathrm{OPT}_{\mathrm{LP}} = \sum_j \lambda_j w_j \rho_j$. For the fairness half, (30) gives $\rho_j^{\mathrm{ALG}} = \lim_{T\to\infty} S_j^{\pi,T}/\lambda_j \geq (1 - \alpha_j^{\mathrm{block}})(\rho_j - \delta_j)_+$, from which the rest of the inequality sequence follows. □

Proof of Theorem 3. At the clock $u_j = \log\kappa / D_j$ and the scaling $\lambda_j = \kappa \lambda_j^\circ$ of Definition 2, (33) gives

$$\alpha_j^{\mathrm{block}} \;\leq\; \frac{u_j}{\lambda_j \bar{K}_j^{\mathrm{don}}} \;=\; \frac{\log\kappa}{\kappa\, \lambda_j^\circ \bar{K}_j^{\mathrm{don}} D_j} \;=\; O\Big(\frac{\log\kappa}{\kappa}\Big),$$

while $u_j D_j = \log\kappa$, so with G_j fixed (22) gives

$$\delta_j \;\leq\; \mathbb{P}\big[\mathrm{Poisson}(\log\kappa) < G_j\big] \;=\; O\Big(\frac{(\log\kappa)^{G_j - 1}}{\kappa}\Big).$$

Both are $\tilde{O}(1/\kappa)$. Finally $\mathrm{CR} = (1 - \alpha_j^{\mathrm{block}})(1 - \delta_j/\rho_j)_+ \geq 1 - \alpha_j^{\mathrm{block}} - \delta_j/\rho_j = 1 - \tilde{O}(1/\kappa)$ as ρ_j remains constant under the scaling of Definition 2; taking $\min_j$ over the fixed $|\mathcal{J}|$ types gives $\mathrm{CR} = 1 - \tilde{O}(1/\kappa)$. □

Proof of Proposition 2. *(i) FIFO is offline-optimal.* Write s_p for project p's arrival time and $e_p := s_p + D$ for its deadline. Any policy's realized outcome is a feasible matching of funded projects to donor epochs, project p matched to a donor epoch d only if $d \in [s_p, s_p + D]$. Since all windows have equal length D, oldest-first coincides with earliest-deadline-first, and the classical exchange argument gives a maximum matching agreeing with FIFO at every donor epoch: if the optimum leaves an epoch unused, advance FIFO's choice into it; if it serves $p' \neq p$ with $e_p \leq e_{p'}$, swap the two assignments, which stays feasible because the later epoch still precedes $e_{p'}$. (If the optimum serves p' at d but never serves p, reassign d to p and leave p' unserved; the cardinality is unchanged.) Applying the exchange at each donor epoch in increasing time order, and noting that FIFO's decision at an epoch does not depend on the horizon, gives $F_\pi(T) \leq F_{\mathrm{FIFO}}(T)$ simultaneously for every T.

(ii) FIFO's exact loss. Let $W \in [0, D]$ be the virtual waiting time: it decreases at unit rate and, at each arrival, jumps by $\min(E, D - W)$ with $E \sim \mathrm{Exp}(\mu)$ fresh — an arrival waits W to reach the head, then a fresh donor gap, and is lost precisely when $E > D - W$. Let us guess the form of the stationary distribution of this virtual

waiting time as $p_0\delta_0(w)+g(w)$, where $\delta_0(w)$ is the Dirac delta function and the scalar p_0 and function $g(\cdot)$ are to be found. With this in mind, for the distribution to remain stationary, it needs to equate the downcrossing rate $g(w)$ with the upcrossing rate $\lambda\big[p_0e^{-\mu w}+\int_0^w g(x)e^{-\mu(w-x)}dx\big]$. The constant function $g\equiv\mu p_0$ solves this exactly when $\lambda=\mu$: the $e^{-\mu w}$ terms cancel iff $g=\mu p_0$ and the remaining terms iff $\lambda=\mu$. So W is uniform on $(0,D)$ with an atom at 0, and $p_0(1+\mu D)=1$. By PASTA an arrival sees $W\sim\pi_W$, so

$$\mathbb{P}[\text{lost}]=\mathbb{E}\big[e^{-\mu(D-W)}\big]=p_0e^{-\mu D}+\int_0^D \mu p_0e^{-\mu(D-w)}dw=p_0=\frac{1}{1+\mu D}.$$

(iii) Combining, $J(\pi)\le\mu(1-p_0)$ and $\mathrm{OPT}_{\mathrm{LP}}=\mu$ at $\lambda=\mu$. Under Definition 2 $\mu=\kappa\mu^\circ$ with D fixed, so $p_0=1/(1+\mu D)=\Theta(1/\kappa)$. □

EC.3.1. Proofs for the sponsor-capital section

This appendix collects the proofs deferred from Section EC.2, in the order in which the statements appear. The transfer of the product form to the enlarged phases requires constructing the embedded kernel on those phases and re-verifying the partial-balance identity the product-form proof consumes; that construction sits between the two lemmas it serves.

Proof of Theorem EC.1. We follow the proof of Proposition 1: fix T, build empirical rates, show they are feasible for (EC.2) at every T, and take $\limsup$ only at the end. Fix T and set

$$M_j^{\pi,T}(g,k):=\tfrac{1}{T}\,\mathbb{E}_\pi\big[\#\{\text{type-}j\text{ matches at }(g,k)\text{ by }T\}\big],$$

and likewise $N_j^{\pi,T}((g,k)\to h)$, $\Pi_j^{\pi,T}(g,k)$ for entries, and $\Phi_j^{\pi,T}(k)$ for fundings with accumulated capital k. Only three steps differ from that proof.

Capacity. Payments are at least one unit and capital jumps go strictly upward, so a project's level strictly increases and k never decreases; the project therefore enters each state (g,k) at most once and takes at most one action there, a donor match or a capital jump. Summing over projects gives $M_j^{\pi,T}(g,k)+\sum_h N_j^{\pi,T}((g,k)\to h)\le\Pi_j^{\pi,T}(g,k)$ at every T, since the inequality holds project by project, and $\Pi_j^{\pi,T}(0,0)=\lambda_j$, an arrival the policy rejects appearing as slack in the capacity constraint.

Entry recursion. Entries to (g,k) come from a donor match at (r,k), $r<g$, whose payment is $g-r$ (matches do not change k), or from a capital jump out of $(r,k-(g-r))$, which involves no randomness. Since $q_{j,r}$ depends on the level alone, the first are counted by the Wald argument of Proposition 1, giving the entry recursion of (EC.2) exactly at every T. The donor constraint and the objective follow as there too, since capital jumps route no donor: $(\sum_{g,k}M_j^{\pi,T}(g,k))_j\in\widehat{F}_+$ and $\sum_j w_j\sum_k\Phi_j^{\pi,T}(k)=\frac{1}{T}\mathbb{E}_\pi[\text{value earned by }T]$, completions by a capital jump to G_j being counted exactly by the definition of $\Phi_j^{\pi,T}$, as in the second sum of (EC.1).

Budget, and the limit. The platform disburses a project's capital exactly when the project completes, in the amount the second coordinate of its state records, so $\Psi^{\pi,T}=\sum_j\sum_k k\,\Phi_j^{\pi,T}(k)$ identically at every T, and budget-feasibility gives $\limsup_T\Psi^{\pi,T}\le B$. This is the one constraint that need not hold at finite T, so we pass to a limit. Every empirical rate is bounded by $\sum_j\lambda_jG_j$, since each project contributes at most G_j

events of each kind, its level strictly increasing at every transition (see *Capacity*), so along $T_n \to \infty$ with the objective converging to $J(\pi)$ we may refine by Bolzano–Weierstrass until every coordinate converges. The other constraints are closed and pass to the limit, and Ψ is linear in the rates, so the limit is feasible for (EC.2) with the fairness constraints deleted, with value $J(\pi)$, and $J(\pi) \le \mathrm{OPT}^{\mathrm{enl,unc}}(B)$. For a fair policy the floors pass to the limit as in Proposition 1, which gives the second claim. □

Proof of Lemma EC.1. Adopt the convention $c_j(g,k) := 0$ and $\sigma_j((g,k) \to h) := 0$ at states with $\Pi_j(g,k) = 0$; no project ever enters such a state, so the choice is immaterial and it removes the $0/0$ ambiguity. The capacity constraint of (EC.2) divided by $\Pi_j(g,k)$ gives $c_j(g,k) + \sum_h \sigma_j((g,k) \to h) \le 1$ at every state carrying flow, so the coins and jump shares define a sub-probability and the residual mass is absorption at $\star$.

Entries to (g,k) happen through quantities at states of strictly smaller level g: donor matches contribute from (r,k) with $r < g$, and capital jumps from $(r, k-(g-r))$ with $r < g$. The system is therefore lower-triangular in g. The identification $\Pi_j = \lambda_j \hat{b}_j$ and $M_j = \lambda_j \hat{a}_j$ follows by induction on g. It holds at $(0,0)$ because $\Pi_j(0,0) = \lambda_j$ and $\hat{b}_j(0,0) = 1$, and if it holds at every level below g then substituting into the two recursions gives identical right-hand sides, so it holds at $(g,\cdot)$.

For the absorption statement, condition on the completing transition. A project at some level g whose eventual total capital consumption is k must be funded either by a donor match at (g,k) whose payment overshoots, of probability $\hat{a}_j(g,k)\bar{q}_{j,g}(G_j - g)$, or by a jump to the goal out of $(g, k-(G_j-g))$, of probability $\hat{b}_j(g, k-(G_j-g))\,\sigma_j((g,k-(G_j-g)) \to G_j)$. Each project is funded once, by exactly one transition, so these events are disjoint, and their λ_j-multiples are precisely the two sums in (EC.1). Hence $\lambda_j \mathbb{P}[\text{funded with } k] = \Phi_j(k)$, and summing over k gives $\rho_j^{\mathrm{enl}} = S_j/\lambda_j$ for the absorption probability at G_j. The expected capital spent per arrival is then $\sum_k k\,\mathbb{P}[\text{funded with } k] = \lambda_j^{-1} \sum_k k\,\Phi_j(k)$. □

In order to get the stationary distribution of the system state under our policy, we need an extra step when converting the DTMC constructed above to a CTMC that captures the actual time spent at each state. In particular, note that the capital jumps in this extended model happen immediately upon entering a state. This means that even though the DTMC's transitions consist of both types: donor-match and capital jump, the latter happens instantaneously and needs to be absorbed into the former type transitions, resulting in a pure donor-match embedded DTMC, which can then be treated the same way as in Section 4 to construct the CTMC, analyze the deadline tail, and provide the stationary count distribution and blocking probability. The following definition formalizes this.

DEFINITION EC.3 (DONOR-ACTIVE ENLARGED PHASES AND THEIR KERNEL). A project is *donor-active* at (g,k) if it has entered (g,k) and its continuation coin $c_j(g,k)$ is realized, so that it awaits a donor consumption tick at (g,k). On the other hand, for an entry to (g,k) whose realized action is a capital jump, the transition is instantaneous: the project moves to $(h, k+h-g)$ and again draws its action there, possibly jumping again, until it becomes donor-active, is funded, or is absorbed at $\star$. Jumped-through transitions therefore impose zero occupancy on their underlying states over time and are not phases of the Poisson count process.

Write $\nu_j(\cdot)$ for the initial distribution of the first donor-active phase of an arrival, that is, the resolution of the cascade out of $(0,0)$, and $r_j := \sum_{(g,k)} \nu_j(g,k)$ for the probability that an arrival ever becomes donor-active (the rest immediately absorb at either G_j, funded fully by capital, or at $\star$, discarded). Define the *enlarged embedded kernel* $\widetilde{P}^{(j)}$ on donor-active phases by composing one donation step with the ensuing instantaneous cascade,

$$\widetilde{P}^{(j)}_{(g,k),(h,l)} := \mathbb{P}\big[\text{the next donor-active phase is } (h,l) \;\big|\; \text{donor-active at } (g,k)\big],$$

and let $\widetilde{q}^{(j)}_{(g,k)} := 1 - \sum_{(h,l)} \widetilde{P}^{(j)}_{(g,k),(h,l)}$ be the absorption probability, success or $\star$.

Let us first show the following identity (EC.9) which is the analogue of (EC.8), the identity the product-form proof of Theorem 1 consumes, with two minor changes. The phase index is (g,k) rather than g, and the entry term is the cascade distribution ν_j rather than a point mass $a(0)$ at level $g=0$.

LEMMA EC.3. *With $\hat{a}_j$ as in Lemma EC.1 and $\widetilde{P}^{(j)}, \widetilde{q}^{(j)}, \nu_j$ as in Definition EC.3,*

$$\sum_{(g,k)} \hat{a}_j(g,k)\, \widetilde{P}^{(j)}_{(g,k),(h,l)} \;=\; \hat{a}_j(h,l) - \nu_j(h,l) \qquad \textit{for every phase } (h,l), \tag{EC.9}$$

and $\sum_{(g,k)} \hat{a}_j(g,k)\, \widetilde{q}^{(j)}_{(g,k)} = r_j$.

Proof. The quantity $\hat{a}_j(h,l)$ is the probability that an arrival is ever donor-active at (h,l). The level strictly increases, so the phase is visited at most once and no multiplicity correction arises. Decompose that event on how the phase was first reached. Either (h,l) is the arrival's first donor-active phase, of probability $\nu_j(h,l)$, or the project was donor-active at some earlier phase (g,k) and (h,l) was the next one, of probability $\hat{a}_j(g,k)\widetilde{P}^{(j)}_{(g,k),(h,l)}$ summed over (g,k). The two cases are disjoint and cover the event, which is (EC.9). For the second identity, every arrival that ever becomes donor-active is eventually absorbed, since the chain is upward-monotone with at most G_j donor-consuming steps, so the total absorption flow equals the total entry flow r_j. □

Proof of Lemma EC.2. The proof of Theorem 1 holds for a general per-label chain: it uses the chain only through the identities (EC.8), which Lemma EC.3 supplies on the enlarged phases with two changes, the phase (g,k) in place of g and the cascade distribution ν_j in place of entry at level 0. An arrival therefore occupies a slot at rate $\lambda_j r_j$ in place of $\lambda_j a_j(0)$, and with phase weights $\hat{a}_j(g,k)/\bar{K}_j^{\mathrm{don,enl}}$ in place of $p_{j,g}$ the partial-balance verification goes through (phases with $\hat{a}_j = 0$ carry no occupancy and are dropped, as in Lemma EC.1). The entry distribution enters per phase: at (h,l) the service inflow leaves exactly the deficit $\nu_j(h,l)$ of (EC.9) unfilled, and the arrival inflow supplies precisely that deficit, the two summing to the required total. The count is again truncated geometric and, given the count, the phases are multinomial with these weights. The enlargement leaves the count vector, and hence the admission region, untouched, and $\bar{y}_j^{\mathrm{enl}} = \lambda_j \bar{K}_j^{\mathrm{don,enl}}$ by (EC.3) gives $\varrho_j = 1$, so the count is uniform on the region and on the box $\mathcal{K}^{\square}$ the blocking is $1/(K_j^{\square}+1)$. The stationary initialization is seeded accordingly, with initial phases drawn from the visit shares.

Two points have no counterpart in Theorem 1. First, a capital jump routes no donor and displays nothing, so it enters neither side of (19); and cascades are instantaneous, so capital-only successes and pre-donor drops

never occupy a slot, and the donor-active count has the balanced-sharing dynamics under the kernel $\widetilde{P}^{(j)}$ at pooled rate $\bar{y}_j^{\mathrm{enl}}$. Second, insensitivity needs the phase dynamics to be internal to each project: the coins $c_j(g,k)$ and the jump probabilities $\sigma_j((g,k)\to h)$ are drawn on entry to a phase with the project's own randomness, and jumps take no time, so they add no clock that could couple to the queue. □

Proof of Theorem EC.2. Budget feasibility of the policy. The analysis follows those of Theorem 2, Corollary 2, and Theorem 3 for the most part. The main new check we need to do is the budget. By Lemma EC.1 the enlarged chain funds a type-j project with capital k at rate $\Phi_j(k)/\lambda_j$, so its expected capital spent per arrival is $\lambda_j^{-1}\sum_k k\Phi_j(k)$, which (EC.2) caps at B in aggregate over all arrivals. The online policy serves a thinned subset of arrivals, since some are blocked at admission, which would in fact lower the budget consumption compared to the fluid LP's plan. As such, the policy is budget-feasible in the sense of Definition EC.2. Moreover, similar to the guarantees in Section 4, because we seed the system to start at stationarity, this feasibility guarantee holds at any point in time, not just in the long-run limit, and every arrival carries the per-arrival guarantee of Theorem 2 whenever it arrives.

The bound. By Lemma EC.2 the enlarged policy is the policy of Theorem 2 run on the enlarged phase space, so that theorem applies with $(\rho_j^{\mathrm{enl}},\delta_j^{\mathrm{enl}})$ in place of (ρ_j,δ_j), and with $K_j^{\square}$ and $\alpha_j^{\mathrm{block}}$ those of the enlarged box $\mathcal{K}^{\square}$ of Lemma EC.2. It absorbs at G_j with probability ρ_j^{enl} by Lemma EC.1, and its service requirement is at most G_j donations — each donor tick strictly increases the level — so (22) applies with the same G_j and the same deadline-miss analysis, and the bound does not depend on the horizon at all. A project that capital funds or drops on arrival never faces the admission rule, so the count process of Lemma EC.2 and its blocking are unchanged, and a type with $\bar{K}_j^{\mathrm{don,enl}}=0$ earns its planned ρ_j^{enl} in full. Capital-only successes are instantaneous and meet any deadline, so the displayed bound is conservative for those types. Budget-feasibility was shown above. The objective decomposition of Corollary 2 then gives (EC.4) with $\mathrm{OPT}_{\mathrm{LP}}^{\mathrm{enl}}(B)=\sum_j \lambda_j w_j \rho_j^{\mathrm{enl}}$. Additionally, because the enlarged program is homogeneous now in the arrival rates and the budget, when the budget scales alongside the market as in Definition 2, the chains from the optimal plan, including $\bar{K}_j^{\mathrm{don,enl}}$ and ρ_j^{enl}, remain scale-invariant as before, recovering the $\big(1-\tilde{O}(1/\kappa)\big)$ scaling behavior of our CR as in Theorem 3. □

EC.4. Additional numerical analyses

This section collects the details and supporting analyses of the calibrated study of Section 6: the data, calibration and implementation behind it, with the estimated payment kernel (Section EC.4.1); the price of fairness (Section EC.4.2); and robustness to perturbations, observed or with estimation error (Section EC.4.3).

EC.4.1. Data, calibration, and implementation

Data and calibration window. The KDD Cup 2014 release contains every DonorsChoose project posted between 2002 and May 2014, with per-donation records (anonymized donor, timestamp, dollar amount to project) through April 2014 for projects posted before 2014. We calibrate on projects posted in calendar 2013,

the last full year whose four-month funding windows close inside the donation record, and keep posts from June 2012 onward only to reconstruct the live project pool. The 2013 cross-section has 131,304 posts (a total arrival rate of 359.7/day) and 801,958 donations. We validate the reconstruction internally: cumulating donation amounts against each project's goal reproduces the dataset's funded/unfunded labels, with 70.0% funded against the labeled 70.4%. Donations from teacher accounts (16.8% of the $801{,}958$ donations to 2013 posts, mostly own-project seeding) are excluded from the donor-side estimates, and promotion-matched donations (18.4% of the same) are kept and flagged.

Type schema and primitives. Project types are the $J = 42$ cells of (primary focus area) $\times$ (highest-poverty indicator) $\times$ (goal tercile within cell), chosen so that goals are homogeneous within a type, the protected set is meaningful, and every estimation cell remains thick. Donor types are the U.S. states plus an out-of-coverage bucket ($I = 52$). Table EC.1 summarizes the estimated primitives. Goals are discretized at \$20 per level unit. Deadlines are not recorded, but among never-funded 2013 projects the age of the last observed donation has 90th/95th/99th percentiles of $115.6/118.3/122.5$ days, so we set $D_j \equiv 120$ days, which is also the typical campaign period on DonorsChoose reported by Song et al. (2022). Payment pmfs $q_{j,g}$ are estimated per type and per quartile of progress toward the goal (median cell: 1,901 donations; minimum: 247), with amounts capped at $\max_j G_j$. Payments are rounded up to whole \$20 units. Positivity holds by construction of the \$20 unit. The protected set $\mathcal{P}$ consists of the 21 highest-poverty types, which account for 54% of arrivals.

Every per-type minimum and maximum below is read over the served set $\mathcal{J}^\rho$ of Remark 5, or stated with the mass of the type attaining it, because the solved benchmark funds one type at $\rho_j = 0.0021$, and that type attains almost every extremum taken over all types.

Choice model. The choice functions $\varphi_i(j \mid R)$ are multinomial logit (MNL) per donor type, the canonical weakly substitutable family (Remark 1), and the revenue-ordered structure of Talluri and van Ryzin (2004) supplies the assortment-optimization oracle of Lemma 4. The displayed assortment is never recorded in the data, so we use the fraction of donations made toward different types as a proxy to estimate the MNL weight parameters of the choice model. Specifically, we note that every type has at least 35 live projects on every day of 2013, so the maximum-likelihood attraction weights can be approximated via a simple share inversion: $v_{ij} \propto$ the share of donor type i's donations that go to project type j. Under full display these shares identify the MNL attractions up to the outside option, which the data cannot identify, as donation logs record selections and not visits.

We set the full-display conversion rate to $c = 0.90$, i.e., a donor who reaches the platform and observes all types selects some project nine out of ten times. Donor arrival rates are then $\mu_i = (\text{selection rate}_i)/c$, and total donor capacity is held at the observed $1{,}828$ selections/day regardless of c, so the choice of c affects only the substitution between project types. With a small outside option, displaying one type competes with displaying another, so the assortment is a real decision; full display stops being the exact LP optimum once the outside option falls below about 40% of choice mass under the throughput and revenue weightings, and below about 78% under the mission weighting.

We consider three value-weighting schemes, two of which come from Section 2.1: under throughput, $w_j \equiv 1$; under revenue, $w_j = G_j$; and under the mission weighting, w_j is the median number of students that a type-j project reaches (the data record `students_reached` for each project). The revenue and mission weights are rescaled to an unweighted mean of one across the 42 types, so all three schemes value the average project type at one.

Primitive	Source	Estimate
λ_j	posting timestamps	359.7/day total; 1.45–29.89/day per type
G_j	goal (excl. optional support)	type medians \$224–\$909 (11–45 units of \$20)
D_j	last-donation-age truncation	120 days (all types)
$q_{j,g}$	per-donation amounts	per (type, level quartile); ≥ 247 obs/cell
μ_i	donation timestamps, $c = 0.90$	2,031/day total (1,828 selections/day)
$\varphi_i(j \mid R)$	MNL share inversion	full availability (≥ 35 live/type-day)
w_j	—	$\{1, G_j,$ students reached$\}$
$\mathcal{P}$, β_j	poverty level	21 highest-poverty types (54% of arrivals)
empirical $\hat{\rho}_j$	outcome labels	0.55–0.92 across types (arrival-weighted mean 0.70)

Table EC.1 Calibrated primitives, DonorsChoose 2013 ($J = 42$, $I = 52$). The bottom panel reports the empirical funded rates from the outcome labels. They are not model primitives. They anchor the fairness floors (Section 6.6) and give the realized success share that the benchmark is compared with.

The estimated payment kernel. In the following we plot the payment kernel of 4 different project types given what quartile their level lies in. The mean donation is smallest in the top quartile of all four types, at 1.57, 2.04, 2.77 and 2.48 units, against first-quartile means of 3.07, 4.45, 8.04 and 5.77. Donations are smallest near the goal, though they do not fall monotonically across the quartiles. This supports allowing the payment to be level dependent, which is carried by $q_{j,g}$ in the model. Mass beyond the dashed goal line is expected: support runs past G_j in 41 of the 42 types, and it is the overshoot that the success probability $\bar{q}_{j,g}(G_j - g)$ integrates.

Computing the benchmark. On the calibrated instance, the program (P) of Section 3.1 has 3,192 variables, with Donor Availability (6) written in the sales-based form of the MNL choice polytope (Gallego et al., 2015). It is solved in a few seconds by a structured interior-point method (Mehrotra, 1992), and every solution is certified by primal, dual and gap residuals below 10^{-6}. The plan the policy runs is a basic optimal solution of the same program, used only if its match rates M agree with those of the interior-point solution to a relative tolerance of 10^{-5}. Where the split of donor flow across donor types is not unique, a fixed generic linear tie-break picks one, and the policy's value does not depend on the split. With $w_j \equiv 1$, $\mathrm{OPT}_{\mathrm{LP}} = 279.0$ funded projects/day against 359.7 arrivals/day, i.e., an optimal per-arrival success share of 77.6%, compared with the platform's realized 70.4%. The seven-point gap is the headroom the model finds in the platform's operation in the 2013 data, and about two points of it come from choosing the assortment: at a 90% outside share, where full display is optimal, the gap is five points. The structural quantities are read over $\mathcal{J}^{\rho}$ of Remark 5, which at throughput weights is 37 of the 42 types. Over this set, $\rho_j \in [0.002, 1.000]$, with the bottom of the range set by types the benchmark has effectively given up on. A side observation here is that Panel (d) of Figure EC.1 is the type the solved throughput benchmark nearly abandons, at $\rho_j = 0.0021$. It is also the third heaviest type in the data by donation count, at 50,277 donations, which highlights how the optimal plan might be very different

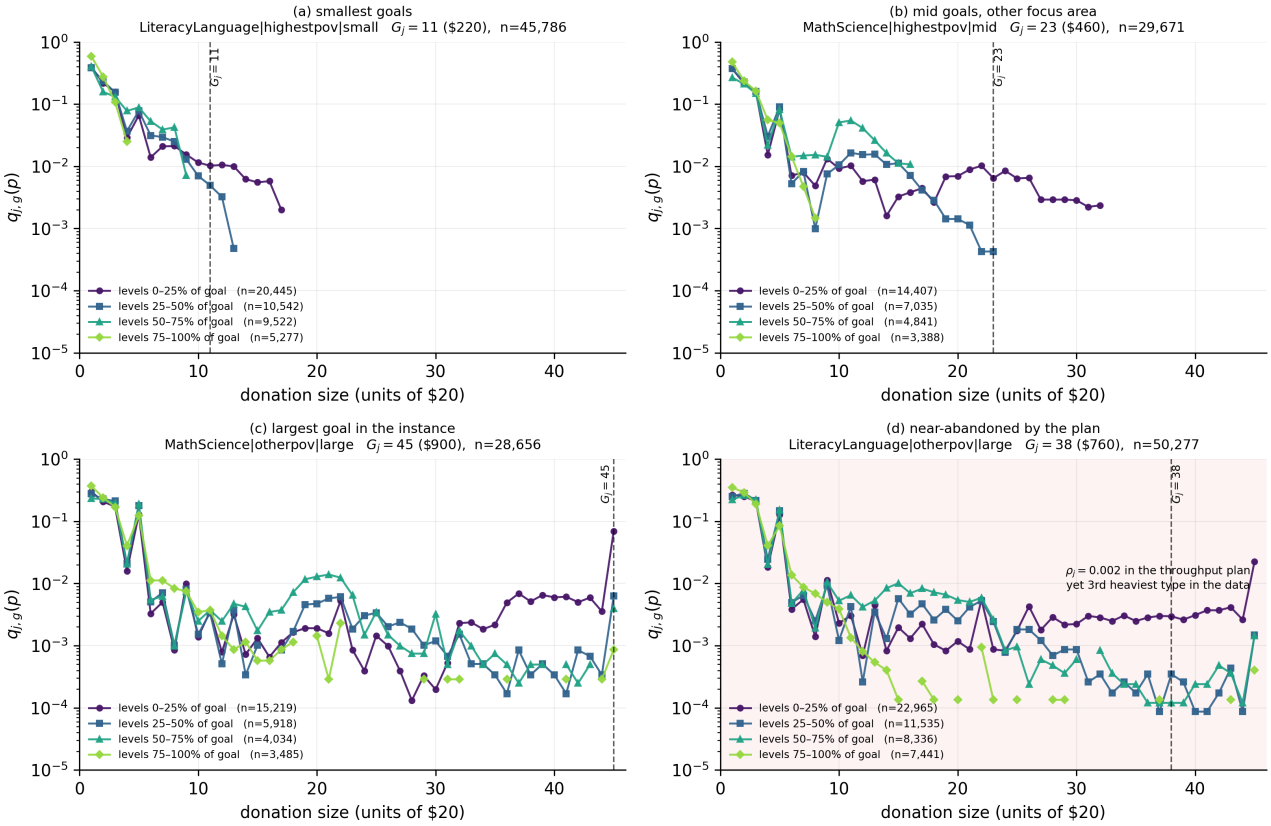


Figure EC.1 The estimated payment kernel, $q_{j,g}$, for four types spanning the goal range, one panel each, with levels grouped into quartiles of the goal and every series labeled by its own observation count. Panel (a) is the smallest goal tier, (b) a mid tier in a different focus area, (c) the largest goal in the instance, and (d) the type the solved throughput plan nearly abandons. Each panel title names the type as focus area | poverty group | goal tercile, where `highestpov` marks the schools DonorsChoose classifies as highest-poverty and `otherpov` all other schools.

from what the platform is currently doing. We note that the plan also depends heavily on the weighting: only 17 of the 42 types are served under all three weightings, and 24 of the other 25 are abandoned under some weighting but funded at 0.95 or more under another. Throughput funds every small type and drops five large ones, revenue drops every small type, and the mission weighting drops the types whose donations reach the fewest students, the six Special Needs types among them; this is also why it drops four of the types donors pick least often (Section EC.4.2). The type of Panel (d), for instance, is funded fully under revenue and not at all under the mission weighting.

Implementation of the policy, and unit verification. Over the same set, $\bar{K}_j^{\mathrm{don}} \in [0.02, 7.59]$, its bottom again set by those types; weighted by funded rate, the 5th to 95th percentile of $\bar{K}_j^{\mathrm{don}}$ is $[4.48, 7.11]$ (Figure EC.2). Curate-Dedicate uses the box $\mathcal{K}^{\square}$ with $K_j^{\square} = \lfloor \bar{y}_j / u_j \rfloor$, which downward closure places inside the feasible region automatically (Lemma 4), with the exact truncated-Poisson admission probability. Initialization follows (I1)–(I3), except that the initial count is drawn from Curate-Dedicate's own stationary law on $\mathcal{K}^{\square}$, a truncated Poisson distribution, in place of that of (I2). The reported runs use the loss-network reduction of Remark 8, which is exact under Curate-Dedicate's per-project clocks, so the runs simulate admissions and

sojourns and evaluate per-label success in closed form. The reduction, the display distribution and the admission identity are verified on an enumerable instance and against independent donor-stream simulators, one for Curate-Dedicate and two for Curate-Share. Curate-Share is simulated through the donation law its display induces: by (19), while type j has an active label, its donations arrive as a Poisson stream of rate $\bar{y}_j$, and (O2) routes each donation to a uniformly random active label. Every reported ratio is a long-run rate. For Curate-Share, type j's success per arrival is its exact admission probability $K_j^{\square}/(K_j^{\square}+1)$ (the count is uniform on $\mathcal{K}^{\square}$) times the funded fraction of its admitted projects, pooled over replications; when capital is present, the projects that capital funds on arrival (without going through admission) are earned without loss. The policy runs start in stationarity, and the greedy baseline runs burn in from empty, i.e., they start from an empty platform and are measured only after a burn-in of $4{,}000$ days ($12{,}200$ for top-12 display, and 900 when the donor supply is scaled down).

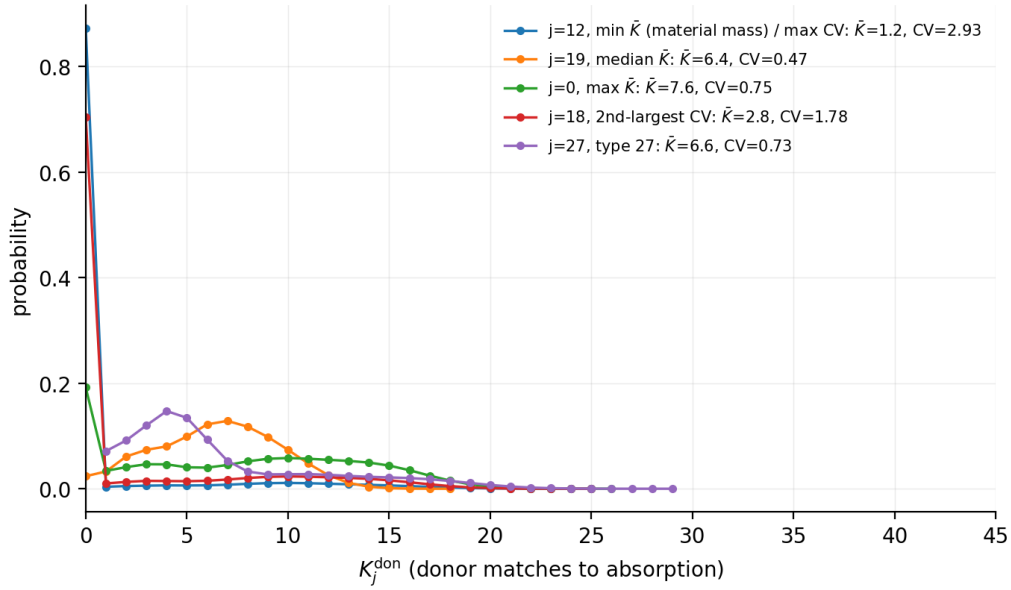


Figure EC.2 The donor-effort distributions. Exact pmf of K_j^{don} for five representative types at the calibration used here, under throughput weights: the smallest, median and largest $\bar{K}_j^{\mathrm{don}}$, the second-largest coefficient of variation, and type 27. The horizontal axis runs to 45 because $K_j^{\mathrm{don}} \le G_j$ and the largest calibrated goal is 45 level units.

EC.4.2. The price of fairness

A protected set that donors disfavor is the sharper test of the guarantee of Corollary 2. We redefine the protected set as the ten types whose projects donors pick least often, i.e., with the fewest 2013 donations per live project-day (0.073–0.080, against a median of 0.085), where a project is live from its posting until it is funded or reaches its deadline. They are five mid and five large types, funded at 55–70% in 2013, and under the mission weighting we impose floors $\beta \equiv 0.733\,\gamma$ on them, where 0.733 is the unweighted mean 2013 funded rate of the other 32 types. Without floors, the plan drops four of them: the three Special Needs types in the set and

large Literacy projects in the highest-poverty schools, which reach only 1.8–3.1 students per donation, against at least 3.7 for every type the plan funds. The floors restore them, at a price of 0.26, 0.55, 0.88, 1.73 and 3.03% at $\gamma = 0.15$, 0.3, 0.45, 0.75 and 1, with 4, 4, 5, 6 and 7 of the ten floors binding.

Full display still reaches these types, since any displayed type has a positive selection probability, while display restriction collapses the types it does not show to rate zero; what full display cannot do is *steer*. The greedy rules ignore the floors: almost-funded-first funds the worst-off of them at 0.59 and FCFS at 0.45. Without floors, Curate-Share follows the plan and funds four of them at zero. With the floors, Curate-Share at its searched caps funds the worst-off at 0.10, 0.21, 0.31, 0.54 and 0.72 at the five levels, that is, at 94–99% of the floor; at parity, it serves the worst-off better than almost-funded-first, at 0.72 against 0.59, with a protected-group mean of 0.78 against 0.65 and a competitive ratio of 0.995.

With sponsor capital, given alike to the fair and the unconstrained programs, the price falls with the budget under every weighting and is zero from $B^\star$ on, where both fund every project (Table EC.2).

weighting	γ	$B=0$	2%	5%	10%	$B^\star$
throughput	0.5	0.07	0.02	0.00	0.00	0
	0.75	0.20	0.04	0.00	0.00	0
	1.0	0.47	0.07	0.01	0.00	0
	1.2	2.56	0.92	0.54	0.02	0
revenue	0.5	0.90	0.56	0.16	0.01	0
	0.75	1.40	0.90	0.38	0.01	0
	1.0	1.97	1.33	0.65	0.02	0
	1.2	2.52	1.73	0.90	0.03	0
mission	0.5	0.82	0.36	0.17	0.13	0
	0.75	1.42	0.73	0.27	0.21	0
	1.0	2.35	1.16	0.45	0.28	0
	1.2	3.60	1.58	0.81	0.35	0

Table EC.2 The price of fairness with sponsor capital. The value the floors cost, in % of the unconstrained optimum, with the fair and the unconstrained programs given the same budget B, a share of the full-funding cost; at $B^\star = 16.62\%$, both fund every project.

When the floors fall instead on the ten types donors pick least often, they cost the most under the mission weighting, which weights these types below the rest, and capital lowers that price too: from 0.88% with no capital to 0.45%, 0.27% and 0.21% at 2%, 5% and 10% of full funding at $\gamma = 0.45$, and from 3.03% to 1.53%, 0.80% and 0.51% at parity, reaching zero only at $B^\star$.

EC.4.3. Robustness to perturbations: observed or with estimation error

In Figure EC.3, donor capacity is the same in both calibrations; only substitution between project types differs. At a 90% outside share, full display is the exact LP optimum, so almost-funded-first sits within a point of the frontier and slightly above Curate-Share at its searched caps. At 10%, the assortment is a real decision, the greedy rules fall behind, and the gap widens with the weighting.

Every primitive is estimated, so we perturb the world one primitive at a time and measure realized value against the *re-solved* true-world optimum, i.e., the LP of Section 3.1 solved on the perturbed primitives, with

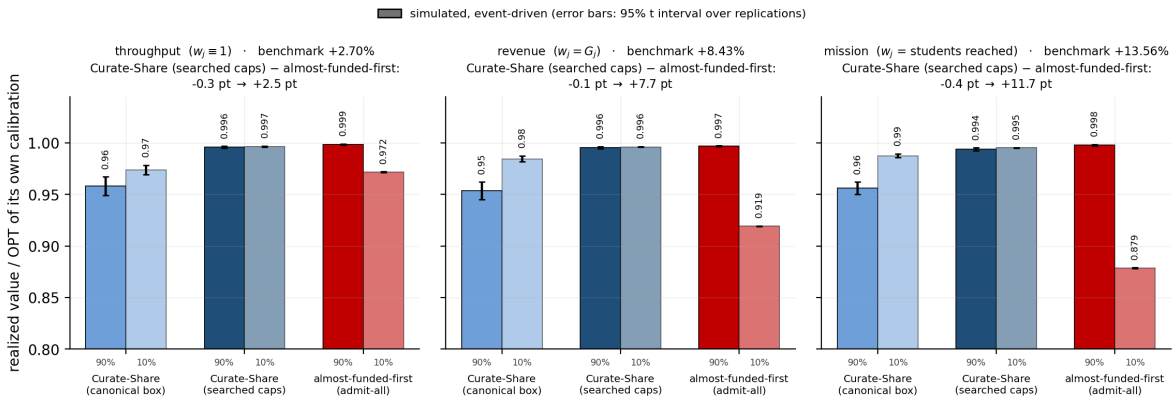


Figure EC.3 Both calibrations, three policies, three weightings. Realized value over the LP optimum at the calibration used here (outside share 10%) and at a 90% outside share. Above each panel are the rise of the LP optimum benchmark from the 90% to the 10% outside share, and the lead of Curate-Share at its searched caps over almost-funded-first, at 90% and then at 10%.

$w \equiv 1$ and no floors. The policy is Curate-Share at its searched caps, run two ways (Table EC.3). The right column studies the robustness of our results to change of primitives, when these perturbed primitives are still known to the policy, and the policy also uses the plan of the LP of Section 3.1 solved on the perturbed primitives, which demonstrates very robust performance. The left column, on the other hand, considers the scenario of estimation error, where the policy builds its plan based on the LP solution with the old calibrated parameters, while the actual realized instance is perturbed. We then record the performance of our policy (which utilizes the erroneous parameters) on the real instance, against the re-solved LP objective which upper-bounds the best policy on the real instance. First, note that a mis-estimated deadline costs almost nothing. The benchmark does not move, since the fluid program of Section 3.1 has no deadline in it, and a deadline 20% shorter than planned lowers the ratio by 0.05 ± 0.05 point.

On the other primitives the experiment is the evidence. Planned on the base instance, a 20% error in the arrival rates, alternating in sign across types, costs 8.49 ± 0.06 points, a donor mass 20% lower costs 4.52 ± 0.07, and making every gift one \$20 unit smaller, which lowers the average gift by 17.5%, costs 4.07 ± 0.07.

true world	estimation error	observed
base (unperturbed)	—	0.996
deadline 20% shorter	0.996	0.995
donor mass 20% lower	0.951	0.996
arrival rates 20% off, alternating up and down across types	0.911	0.996
every gift is one \$20 unit smaller, lowering the average gift by 17.5%	0.956	0.996

Table EC.3 Robustness of Curate-Share ($w \equiv 1$, the calibration is the base here). The base row, 0.996, comes from these runs; the headline runs of Section 6.2 give 0.997 for the same policy and caps.