Outreach Time Intelligence

Adaptive Send-Time Optimization via Thompson Sampling

Dodo Engineering

March 2026

Abstract

Automated outbound campaigns—SMS reminders, reactivation texts, follow-ups—are a core engagement channel for businesses. A common pattern is to schedule all messages at the earliest permissible time, typically when the business opens. This creates a concentrated burst of outreach and ignores per-business variation in when clients are most responsive. We describe an adaptive send-time optimization system based on **Thompson Sampling**, a well-studied multi-armed bandit algorithm. The system discretises the outreach window into 1-hour time slots, maintains a Beta posterior per slot per business, seeds new businesses with global priors computed from historical engagement data, and progressively shifts traffic toward high-performing hours as evidence accumulates. This document covers the mathematical foundations, the design choices behind the system, and the convergence properties that make it practical at real-world engagement rates ($\sim$15%).

1 Motivation

1.1 The Status Quo

Most outbound messaging systems schedule campaigns at a fixed time—often the earliest allowed hour, such as 9 AM in the business's time zone. When a campaign targets hundreds of clients, every message lands simultaneously.

This approach has three problems:

1. **Attention competition.** All recipients are reached at the same moment, competing with each other—and with every other business using the same strategy.
2. **Missed optimal windows.** One business's clients may respond best at 11 AM; another's at 3 PM. A fixed schedule never discovers this.
3. **No learning.** The system does not adapt based on what has worked before.

1.2 What We Want

An ideal system would:

- **Explore** broadly when we have little data, spreading messages across the day.
- **Exploit** the best-performing time slots as evidence accumulates.
- **Personalise** per business, since responsiveness patterns are highly entity-specific.
- **Bootstrap** new businesses using aggregate historical data, rather than starting from scratch.

This is a textbook *exploration vs. exploitation* problem, and there is a rich family of algorithms designed exactly for it.

2 Background: The Multi-Armed Bandit

The name comes from a gambler facing a row of slot machines ("one-armed bandits"), each with an unknown payout probability. The gambler must decide which machines to play to maximise total winnings—balancing the desire to learn about unfamiliar machines (exploration) against the desire to play the best-known machine (exploitation).

2.1 Formal Setup

A **multi-armed bandit** (MAB) consists of K arms, each with an unknown reward distribution. At each round $t = 1, 2, \ldots$:

1. The agent selects an arm $a_t \in \{1, \ldots, K\}$.
2. The agent observes a reward $r_t \sim \nu_{a_t}$, where ν_k is the (unknown) reward distribution of arm k.

The objective is to minimise **cumulative regret**—the total reward lost by not always playing the best arm:

$$R_T = \sum_{t=1}^{T} \big(\mu^* - \mu_{a_t}\big),$$

where $\mu^* = \max_k \mu_k$ is the mean reward of the optimal arm.

2.2 Our Setting: Bernoulli Bandits

In send-time optimization, each arm is a time slot and each "pull" is sending a message in that slot. The outcome is binary:

- $r = 1$: the client *engaged* (e.g., responded within 24 hours).
- $r = 0$: the client did *not* engage.

The reward distribution for arm k is therefore Bernoulli(θ_k), where $\theta_k \in [0, 1]$ is the unknown engagement probability for that time slot.

2.3 Classic Approaches

Three families of algorithms dominate the MAB literature:

Algorithm	**Strengths**	**Weaknesses**
ε-greedy	Simplest to implement; easy to reason about	Wastes exploration budget uniformly across all arms (including clearly bad ones); requires tuning and decaying ε
UCB-1	Deterministic; strong theoretical guarantees	Requires a confidence parameter; less natural for batched or delayed feedback
Thompson Sampling	No hyperparameters; naturally balances exploration and exploitation; data-efficient at low base rates; minimal storage	Requires sampling from a posterior (trivial in practice for the Beta distribution)

Table 1: Comparison of bandit algorithm families.

For our application—low engagement rates ($\sim$15%), binary outcomes, per-entity learning—Thompson Sampling is the clear winner. The rest of this document focuses on it.

3 Thompson Sampling

3.1 Intuition

Thompson Sampling (TS), introduced by Thompson (1933) and recently shown to be asymptotically optimal (Agrawal & Goyal, 2012; Kaufmann et al., 2012), takes a Bayesian approach. Rather than maintaining a single point estimate of each arm's reward, it maintains a full *posterior distribution*. At each decision point, TS **samples** from every arm's posterior and selects the arm whose sample is highest.

The elegance is in how this naturally balances the trade-off:

- **Uncertain arms** have wide posteriors $\Rightarrow$ their samples are spread out $\Rightarrow$ they occasionally win, so they get explored.
- **Well-understood arms** have narrow posteriors $\Rightarrow$ good arms consistently produce high samples $\Rightarrow$ they get exploited.
- As data accumulates, posteriors narrow, and exploitation naturally dominates—without any explicit schedule or decay parameter.

3.2 The Beta–Bernoulli Model

For binary outcomes, the conjugate prior is the Beta distribution, giving us closed-form posterior updates.

Definition 1 (Beta Distribution). *A random variable $X \sim \text{Beta}(\alpha, \beta)$ has probability density*

$$f(x;\, \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}, \quad x \in [0, 1],$$

where $B(\alpha, \beta) = \frac{\Gamma(\alpha)\,\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ is the Beta function.

Key properties:

$$\mathbb{E}[X] = \frac{\alpha}{\alpha + \beta}, \qquad \text{Var}(X) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

The mean $\frac{\alpha}{\alpha+\beta}$ can be read as "fraction of successes so far" (when α counts successes and β counts failures). The variance shrinks as $\alpha + \beta$ grows—more data means more certainty.

3.3 Posterior Updates

The update rule is remarkably simple. Starting from a prior:

$$\begin{aligned} \text{Prior:} \quad & \theta_k \sim \text{Beta}(\alpha_k,\, \beta_k) && (1) \\ \text{Observe:} \quad & r \in \{0, 1\} \\ \text{Posterior:} \quad & \theta_k \mid r \sim \text{Beta}(\alpha_k + r,\, \beta_k + 1 - r) && (2) \end{aligned}$$

In plain language:

- Client engaged ($r = 1$): increment α_k by 1.
- Client did not engage ($r = 0$): increment β_k by 1.

No matrix inversions, no gradient descent, no retraining. Just two integer increments per observation.

3.4 The Algorithm

Algorithm 1: Thompson Sampling for Send-Time Selection

Input: K arms (time slots) with posteriors $\text{Beta}(\alpha_k, \beta_k)$ for $k = 1, \ldots, K$; engagement timeout τ (default: 24 hours)

For each message to schedule:

1. **Sample:** For each arm k, draw $\hat{\theta}_k \sim \text{Beta}(\alpha_k, \beta_k)$
2. **Select:** Choose $a = \arg\max_k \hat{\theta}_k$
3. **Schedule:** Send the message at a random time within slot a; record the observation as *pending*

On client response (within τ hours of send):

4. $\alpha_a \leftarrow \alpha_a + 1$ (immediate inline update)

Periodically (background resolution):

5. For all pending observations where $t_{\text{now}} - t_{\text{send}} > \tau$: $\beta_a \leftarrow \beta_a + 1$

Remark 1 (Asymmetric Feedback)**.** *Positive outcomes (α updates) arrive as events and are processed immediately. Negative outcomes (β updates) are the absence of an event and require a periodic sweep to identify. This asymmetry is inherent to the problem, not a limitation of the algorithm.*

Remark 2 (Batch Diversity)**.** *When scheduling multiple messages at once, each draw is independent. The stochastic nature of Thompson Sampling means messages naturally spread across multiple slots—especially during early exploration when posteriors are wide. No explicit diversification logic is needed.*

3.5 Regret Bound

Thompson Sampling for Bernoulli bandits achieves logarithmic regret:

$$\mathbb{E}[R_T] = O\left(\sum_{k:\,\mu_k<\mu^*} \frac{\ln T}{\mathrm{KL}(\mu_k\|\mu^*)}\right),$$

where $\mathrm{KL}(\mu_k\|\mu^*) = \mu_k \ln\frac{\mu_k}{\mu^*} + (1-\mu_k)\ln\frac{1-\mu_k}{1-\mu^*}$ is the Kullback–Leibler divergence between Bernoulli(μ_k) and Bernoulli(μ^*).

This matches the Lai–Robbins information-theoretic lower bound asymptotically, making Thompson Sampling *asymptotically optimal*—no algorithm can do fundamentally better.

4 Applying It to Send-Time Optimization

4.1 Defining the Arms: 1-Hour Slots

We fix the outreach window to $\mathcal{W} = [09{:}00, 20{:}00)$ in local time and discretise it into $K = 11$ hourly arms:

$$\text{Arms} = \{09, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19\}$$

Each arm k corresponds to the hour [k:00, k:59]. When arm k is selected, the actual send time is drawn uniformly within the hour:

$$t_{\text{send}} \sim \text{Uniform}(k{:}00,\ k{:}59)$$

This adds micro-randomisation within the slot, preventing all messages in the same slot from arriving at the exact same minute.

Remark 3 (Why 1-hour buckets?)**.** *One-hour intervals are the industry standard for send-time optimization (used by Iterable, Emarsys, and others). Finer granularity (30-minute) doubles the number of arms to 22, spreading data thinner and slowing convergence. Coarser granularity (2-hour, 6 arms) loses meaningful signal— the difference between 9 AM and 10 AM engagement can be significant. One hour strikes the right balance.*

Remark 4 (Why a fixed window?)**.** *Business hours vary across entities, which would change the number of arms per entity and complicate the system. A fixed 9 AM–8 PM window provides a consistent 11-arm setup. This range is also regulatory-safe for SMS in all US states.*

4.2 Scope: Per-Business, Shared Across Campaigns

The bandit operates at the **business level**. All campaigns for a given business share the same posteriors.

Rationale:

- Client responsiveness is primarily driven by the business's client demographics and habits, not by the specific campaign type.
- Pooling data across campaigns accelerates convergence—a single business may run several concurrent campaigns, each with modest volume.
- This keeps storage minimal: 11 rows per business (one per arm), each containing two integers (α_k, β_k).

4.3 Defining Engagement

We define engagement as a binary signal with a configurable observation window:

Definition 2 (Engagement)**.** *A message is considered* ***engaged*** *($r = 1$) if the recipient responds within τ hours of the first outbound message. If no response occurs within τ hours, it is* ***not engaged*** *($r = 0$). The parameter τ is a configurable* ***engagement timeout****, defaulting to 24 hours.*

The timeout τ is a hyperparameter worth tuning per deployment. A shorter window (e.g., $\tau = 12\,\text{h}$) gives faster feedback but may miss legitimate late responders. A longer window (e.g., $\tau = 48\,\text{h}$) captures more responses but delays posterior updates and blurs the time-of-day signal (a response at 10 PM to a 9 AM message may reflect the recipient's evening habits, not the effectiveness of the 9 AM slot).

4.3.1 Dual-Path Outcome Resolution

Positive and negative outcomes are observed through fundamentally different mechanisms:

- **Engagement ($r = 1$):** Detected *inline* at the moment the recipient responds. The system checks whether the response arrived within τ hours of the first outbound message. If yes, α_k is incremented immediately.
- **Non-engagement ($r = 0$):** There is no event for a message that goes unanswered—silence is not an event. A periodic **background resolution job** identifies observations that have exceeded τ hours without a response and increments β_k.

This asymmetry is inherent to the problem: positive signals arrive as events, while negative signals are the *absence* of events and can only be identified by the passage of time. The background job runs periodically (e.g., hourly) and resolves all pending observations older than τ.

4.3.2 Late Responses

A response arriving after τ hours is *not* counted as engagement for bandit purposes—β_k has already been incremented by the background job. This is a deliberate design choice: the bandit optimises for *prompt* responsiveness, which is the signal most correlated with send-time effectiveness. Late responses are still valuable for the business (the conversation proceeds normally), but they do not retroactively update the posterior.

Outreach often consists of multi-step sequences (e.g., initial text, then a follow-up), and the sequence is interrupted as soon as the client responds. We attribute engagement to the time slot of the *first* message, since that is the scheduling decision we are optimising.

5 Bootstrapping with Global Priors

A naïve approach initialises every arm with a uniform prior Beta(1, 1), encoding complete ignorance. This works but wastes early sends on pure exploration when we already have historical data.

5.1 The Idea

We can compute **global priors** from aggregate historical engagement data across all businesses. If we observe that 10 AM historically has an 18% engagement rate across thousands of sends, while 5 PM has 11%, we should encode this knowledge into the starting posteriors for new businesses.

5.2 Computing Global Priors

5.2.1 The Naïve Approach and Its Problem

A first attempt might aggregate raw engagement counts per slot across all historical data:

$$\hat{p}_k^{\text{naive}} = \frac{\text{total engagements in slot } k}{\text{total sends in slot } k}$$

This is problematic when campaign types have very different base engagement rates. A reminder campaign with 30% response rate would dominate the prior over a reactivation campaign with 5%, and the resulting prior would be misleading for both.

5.2.2 Relative Lift Normalization

Instead of asking "what is the absolute engagement rate at 10 AM?", we ask "does 10 AM perform *relatively* better or worse than other times, *within* each campaign?" This removes the campaign-level baseline effect.

For each campaign c:

1. Compute the campaign's overall engagement rate: $p_c = S_c / N_c$
2. For each slot k with sufficient data ($N_{c,k} \geq N_{\min}$, e.g., $N_{\min} = 20$), compute the slot-level rate: $p_{c,k} = S_{c,k} / N_{c,k}$
3. Compute the **lift**: $\ell_{c,k} = p_{c,k} \;/\; p_c$

A lift of 1.0 means "same as the campaign average." A lift of 1.3 means "30% better than average." Crucially, lifts are comparable across campaigns with different base rates.

We then aggregate lifts across campaigns, weighted by per-slot sample size to reduce noise:

$$\bar{\ell}_k = \frac{\sum_c N_{c,k} \cdot \ell_{c,k}}{\sum_c N_{c,k}}$$

Finally, we convert back to an absolute prior using the global average engagement rate $\bar{p}$:

$$\hat{p}_k = \bar{p} \cdot \bar{\ell}_k$$

The global prior for arm k is then:

$$\theta_k \sim \text{Beta}(\alpha_k^{(0)},\ \beta_k^{(0)}) \quad \text{where} \quad \alpha_k^{(0)} = 1 + \kappa \cdot \hat{p}_k, \quad \beta_k^{(0)} = 1 + \kappa \cdot (1 - \hat{p}_k)$$

The hyperparameter κ controls the **prior strength**—how much weight the global data carries relative to a business's own observations.

5.2.3 Why Lift Normalization Works

Consider two campaigns: a reminder campaign (30% overall engagement) and a reactivation campaign (5% overall). Both show elevated engagement at 10 AM:

Campaign	p_c	$p_{c,10\text{AM}}$	$\ell_{c,10\text{AM}}$
Reminders	30%	39%	1.30
Reactivation	5%	7%	1.40

Despite a 32-percentage-point difference in absolute rates, the lifts (1.30 and 1.40) tell a consistent story: 10 AM outperforms the campaign average by ∼30–40%. The aggregated lift captures this time-of-day effect cleanly, without letting the high-volume reminder campaign overwhelm the signal.

Slots with fewer than $N_{\min}$ sends in a given campaign are excluded from that campaign's contribution to avoid noise from small samples. Slots with no historical data across any campaign default to a lift of 1.0 (neutral).

5.3 Choosing κ: Prior Strength

The prior strength κ determines how many local observations are needed to "overpower" the global prior:

κ	**Equivalent to**	**Behaviour**
0	Beta(1, 1)	Uniform prior; pure local learning
10	10 pseudo-observations	Light guidance; local data dominates after ∼50 sends per arm
50	50 pseudo-observations	Moderate guidance; local data dominates after ∼200 sends per arm
100	100 pseudo-observations	Strong guidance; useful when per-business volume is very low

A good default is $\kappa \approx 20$–50. This provides meaningful guidance from global patterns while allowing a business's local data to take over within a few weeks of operation.

5.4 Example

Suppose the global average engagement rate across all campaigns is $\bar{p} = 15\%$, and the aggregated lifts per slot are:

Slot	$\bar{\ell}_k$	$\hat{p}_k = \bar{p} \cdot \bar{\ell}_k$	**Interpretation**
9 AM	0.87	13.0%	Below average (morning rush)
10 AM	1.27	19.0%	Best slot—30% above average
11 AM	1.13	17.0%	Above average
⋮			
5 PM	0.73	11.0%	Well below average

With $\kappa = 30$, the 10 AM arm initialises as:

$$\text{Beta}(1 + 30 \times 0.19,\ 1 + 30 \times 0.81) = \text{Beta}(6.7,\ 25.3)$$

This has mean ≈ 0.21 (close to the lift-adjusted 19%, pulled slightly toward 50% by the +1 terms) and encodes the equivalent of ∼30 observations. After a business accumulates its own 200 sends to this slot, the prior's contribution is diluted to less than 15% of the posterior.

5.5 Why Not Just Use Global Data Directly?

Using the raw global counts (e.g., Beta(1,558, 6,642) for 10 AM) would create an extremely tight prior that local data could never overcome. The scaling factor κ controls how much we trust the global pattern vs. allowing local adaptation. The goal is to *inform*, not to *dictate*.

6 Convergence Analysis

6.1 How Much Data Is Needed?

With $K = 11$ arms and a base engagement rate of ∼15%, the table below estimates convergence at different sample sizes:

Sends/arm	Total sends	Engagements/arm	Confidence level
50	550	∼8	Low—posteriors still wide; heavy exploration
100	1,100	∼15	Moderate—best arm likely identified but exploration still frequent
200	2,200	∼30	Good—posterior std. dev. ≈ 0.024; arms differing by ≥5pp are clearly separated
500	5,500	∼75	High—near full exploitation; fine-grained differences detectable

Table 2: Convergence estimates at 15% base engagement rate.

For a business sending ∼50 messages per day, reaching 200 sends per arm (2,200 total) takes approximately **44 days** (∼6 weeks). With global priors seeding the posteriors, effective exploitation begins much sooner.

6.2 Posterior Precision

After n observations with s engagements on an arm (starting from a uniform prior):

$$\theta_k \mid \text{data} \sim \text{Beta}(1 + s,\ 1 + n - s)$$

The posterior standard deviation is:

$$\sigma = \sqrt{\frac{(1+s)(1+n-s)}{(2+n)^2(3+n)}}$$

At $n = 200$, $s = 30$ (15% engagement rate): $\sigma \approx 0.025$. The 95% credible interval spans ∼0.10, meaning we can reliably distinguish arms that differ by 5 percentage points or more.

6.3 Posterior Evolution: An Example

Consider two arms—Slot A with true engagement rate 20% and Slot B at 10%:

Stage	Slot A	Slot B	Separation
Prior (0 sends)	$\text{Beta}(1, 1)$	$\text{Beta}(1, 1)$	None—identical
After 50 sends	$\text{Beta}(11, 41)$, $\mu = 0.21$	$\text{Beta}(6, 46)$, $\mu = 0.12$	Emerging—posteriors overlap
After 200 sends	$\text{Beta}(41, 161)$, $\mu = 0.20$	$\text{Beta}(21, 181)$, $\mu = 0.10$	Clear—minimal overlap

After 50 sends per arm, Thompson Sampling still frequently explores both slots. After 200 sends, the posteriors are well-separated ($\sigma_A \approx 0.028$, $\sigma_B \approx 0.021$), and TS overwhelmingly selects Slot A.

6.4 Effect of Global Priors on Convergence

When we initialise with global priors (Section 5), a new business starts with a posterior that already reflects aggregate patterns. If the global data correctly indicates that 10 AM outperforms 5 PM, the system begins exploiting this immediately, while still exploring enough to detect business-specific deviations.

The convergence timeline shortens from ∼6 weeks (uniform prior) to an effective ∼2–3 weeks, because the prior provides a head start of κ pseudo-observations per arm.

7 Properties and Guarantees

7.1 Asymptotic Optimality

As established, TS achieves the Lai–Robbins regret lower bound. In practical terms, this means the fraction of messages sent at suboptimal times decreases as $O(\ln T/T)$—the system gets better over time and the "cost of learning" is bounded.

7.2 Automatic Exploration Decay

Unlike ε-greedy, TS does not require an explicit exploration schedule. Exploration decays naturally as posteriors narrow. In the early phase, wide posteriors cause frequent exploration; in the late phase, narrow posteriors cause near-deterministic exploitation. There is nothing to tune.

7.3 Robustness to Non-Stationarity

If engagement patterns shift over time (e.g., seasonally), the posteriors will gradually adapt—though slowly, since older observations carry equal weight. For environments with significant non-stationarity, a sliding window or exponential discounting can be applied to the (α, β) values (see Section 8).

7.4 Concurrency Safety

Since the update rule consists only of atomic integer increments ($\alpha \mathrel{+}= 1$ or $\beta \mathrel{+}= 1$), concurrent updates from parallel processes are safe at the database level using atomic increment operations.

8 Future Extensions

1. **Day-of-week awareness.** Extending arms to (day, hour) tuples yields $7 \times 11 = 77$ arms. At ∼200 sends per arm for convergence, this requires ∼15,400 sends per business. Viable for high-volume entities; the global prior approach would be especially valuable here to accelerate convergence.

2. **Non-stationary adaptation.** Applying exponential decay—e.g., periodically scaling $\alpha_k \leftarrow \gamma\,\alpha_k$ and $\beta_k \leftarrow \gamma\,\beta_k$ with $\gamma \in (0.99, 1)$— allows the system to "forget" old observations and adapt to changing patterns.

3. **Contextual bandits.** Incorporating client-level features (e.g., demographics, days since last interaction) transforms the problem into a *contextual bandit*, enabling per-segment rather than per-business personalization.

4. **Campaign-type specialization.** Different campaign types (e.g., appointment reminders vs. reactivation) may have distinct optimal send times. With sufficient volume, maintaining separate bandit instances per campaign type could capture this.

5. **Continuous time modelling.** Instead of discrete hourly buckets, model the engagement probability as a continuous function of time using Gaussian Process bandits. This is more expressive but significantly more complex.

9 Summary

Aspect	Decision
Algorithm	Thompson Sampling with Beta(α, β) posteriors
Arms	11 hourly slots, [9:00, 20:00) local time
Scope	Per-business, shared across campaigns
Prior	Global priors via relative-lift normalization ($\kappa \approx 20$–50)
Engagement	Binary; client responds within τ h (default 24); τ is configurable
Outcome tracking	α updated inline on response; β updated via periodic resolution
Day-of-week	Deferred (11 arms now, 77 arms in future)
Storage	Two integers per arm per business ($11 \times 2 = 22$ integers total)
Convergence	$\sim$2,200 sends per business for full convergence; faster with global priors

Table 3: Design decisions at a glance.

Thompson Sampling offers an elegant solution to the send-time optimization problem. It adapts automatically from exploration to exploitation, requires only one tunable hyperparameter (τ, the engagement timeout), and is trivial to operate—just atomic integer increments on engagement events. The dual-path outcome resolution (inline for positive signals, periodic sweep for negative signals) reflects the natural asymmetry of the problem: responses are events, but silence is not.

Combined with lift-normalised global priors, new businesses benefit from collective intelligence from day one—without high-engagement campaign types distorting the signal—while retaining full ability to discover their own optimal patterns.

The approach generalises beyond messaging—any system that makes repeated decisions with binary outcomes and per-entity variation is a candidate: notification timing, pricing experiments, channel selection, and more.

References

[1] W. R. Thompson. "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples." *Biometrika*, 25(3/4):285–294, 1933.

[2] S. Agrawal and N. Goyal. "Analysis of Thompson Sampling for the Multi-Armed Bandit Problem." *COLT*, 2012.

[3] E. Kaufmann, N. Korda, and R. Munos. "Thompson Sampling: An Asymptotically Optimal Finite-Time Analysis." *ALT*, 2012.

[4] D. Russo, B. Van Roy, A. Kazerouni, I. Osband, and Z. Wen. "A Tutorial on Thompson Sampling." *Foundations and Trends in Machine Learning*, 11(1):1–96, 2018.

[5] T. L. Lai and H. Robbins. "Asymptotically efficient adaptive allocation rules." *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[6] O. Chapelle and L. Li. "An empirical evaluation of Thompson Sampling." *NeurIPS*, 2011.