University of Passau
Faculty of Computer Science and Mathematics

Chair of Computer Science with a focus on Scalable Database Systems
Prof. Dr. Stefanie Scherzinger

Master's Thesis
in
Computer Science

Smart Paths, Better Privacy: Graph-Optimized Cookie-Based Consent Management

Zoya Asadi
100505

Date:	2025-10-22
Supervisors:	Prof. Dr. Stefanie Scherzinger
Advisor:	Prof. Dr. Harald Kosch

Asadi, Zoya
Spitalhofstraße 68a
94032, Passau

ERKLÄRUNG

Ich erkläre, dass ich die vorliegende Arbeit mit dem Titel " Smart Paths, Better Privacy: Graph-Optimized Cookie-Based Consent Management " selbstständig, ohne unzulässige Hilfe und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel verfasst habe und dass alle wörtlich oder sinngemäß übernommenen Stellen als solche kenntlich gemacht sind.

Mit der aktuell geltenden Fassung der Satzung der Universität Passau zur Sicherung guter wissenschaftlicher Praxis und für den Umgang mit wissenschaftlichem Fehlverhalten vom 25. Juli 2023 (vABlUP S. 186) bin ich vertraut.

Ich erkläre mich mit einer Überprüfung der Arbeit unter Zuhilfenahme von Dienstleistungen Dritter (z.B. Anti-Plagiatssoftware) zur Gewährleistung der einwandfreien Kennzeichnung übernommener Ausführungen ohne Verletzung geistigen Eigentums an einem von anderen geschaffenen urheberrechtlich geschützten Werk oder von anderen stammenden wesentlichen wissenschaftlichen Erkenntnissen, Hypothesen, Lehren oder Forschungsansätzen einverstanden.

....
(Asadi, Zoya)

Translation of German text (notice: Only the German text is legally binding)

I hereby confirm that I have composed the present scientific work entitled " Smart Paths, Better Privacy: Graph-Optimized Cookie-Based Consent Management " independently without anybody else's assistance and utilising no sources or resources other than those specified. I certify that any content adopted literally or in substance has been properly identified and attributed.

I have familiarised myself with the University of Passau's most recent Guidelines for Good Scientific Practice and Scientific Misconduct Ramifications of 25 July 2023 (vABlUP page 186).

I declare my consent to the use of third-party services (e.g. anti-plagiarism software) for the examination of my work to verify the absence of impermissible representation of adopted content without adequate attribution, which would violate the intellectual property rights of others by claiming ownership of somebody else's work, scientific findings, hypotheses, teachings or research approaches.

Supervisor contacts:
Prof. Dr. Stefanie Scherzinger
Chair of Computer Science with a focus on Scalable Database Systems
University of Passau
Email: `Stefanie.Scherzinger@uni-passau.de`
Web: `https://www.fim.uni-passau.de/en/computer-engineering/`

Chair of Distributed Information Systems
University of Passau
Email: `Harald.Kosch@uni-passau.de`
Web: `http://www.uni-passau.de/`

Acknowledgements

I would like to express my sincere gratitude to George Konstantinidis and his research team at the University of Southampton for making their algorithms available and for kindly providing implementation details that were essential for this thesis.

I also wish to thank the OpenCms community for their valuable support in answering my technical questions, which helped me overcome practical challenges during system configuration and integration.

Most importantly, I would like to thank my supervisor, Professor Dr. Stefanie Scherzinger, for her continuous support, insightful guidance, and constructive feedback throughout the preparation of this thesis.

Abstract

Online privacy regulations such as the GDPR and ePrivacy Directive require that users provide informed, explicit consent for the use of cookies. However, many current implementations of cookie consent mechanisms burden users with complex and repetitive decision flows, leading to interaction fatigue and suboptimal privacy outcomes. This thesis presents a graph-theoretic approach to simplifying cookie consent interfaces by minimizing the number of required user actions while preserving compliance and transparency.

We modelled the consent process as a graph, where nodes are interface elements and edges are possible steps a user can take. Based on data from OpenCms, we created three versions of the graph: a baseline version, a version with quick-consent, and a version with quick-consent plus auto-apply and implicit-first-use

Using structural data extracted from a content management system (OpenCms), we construct consent graphs and apply five established path elimination algorithms `RemoveRandomEdge`, `RemoveFirstEdge`, `RemoveMinCuts`, `RemoveMinMC`, and `BruteForce`—adapted from prior work by Konstantinidis et al.

The proposed framework enables the reduction of unnecessary decision paths while maintaining the integrity of user choice and data protection requirements. The results show that simply removing edges is not enough. Quick-consent helps to reduce the number of clicks in many cases, but it still requires an explicit save. The biggest improvement came from auto-apply and implicit-first-use, where most paths became shorter and users had to make fewer consent interactions. This work provides a technical evaluation of a graph-theoretic model that represents and analyzes user consent flows in cookie-based systems. Still, it suggests that graph-based modelling can help to design cookie banners that are easier for users and at the same time respect privacy regulations.

Contents

List of Figures

List of Tables

1 Introduction

Imagine a first-time visitor arrives at a digital library to download an audiobook. A cookie banner appears with several categories (Essential, Functional, Analytics, Marketing) and multiple sub-options. To reject non-essential cookies, the visitor must open "More options" expand categories, toggle several switches, and then confirm. If the visitor later revisits the preferences from another page, the interface may present a slightly different sequence, requiring additional navigation before a valid consent state is recorded. Even to accept only essential cookies, users may face several interactions over recurring screens, which increases cognitive load and the chance of an unintentional opt-in.

This vignette reflects a broader reality. Modern web systems rely on cookies to deliver personalization and analytics, while regulations such as the GDPR [1] and the ePrivacy Directive [2] require explicit, informed consent for non-essential processing. In practice, compliance must be operationalized as valid consent states (e.g., "essential-only," "analytics-allowed," "withdrawn") that are unambiguously reachable through the interface and persist across sessions.

Yet interfaces frequently expose redundant or circuitous paths: users traverse long sequences, revisit options, or face parallel branches that differ only superficially. The difficulty is structural rather than cosmetic. Without an explicit model of the consent journey, there is no principled way to reason about path length, redundant branches, or the precise conditions under which a valid consent state is reached.

This thesis takes a structural perspective. We model the consent journey as a directed graph—UI states as nodes and interactions as edges—and apply path-elimination algorithms to remove redundant branches while preserving legal and functional constraints. In concrete terms, "optimize" here means minimizing the user interactions required to reach any target valid consent state without violating requirements such as granularity, revocability, or the non-disablement of essential cookies. The approach is instantiated in a system (OpenCms), we implement the five algorithms—Remove Random Edge, Remove First Edge, Remove Min-Cuts, Remove MinMC, and Brute Force—and link them to the consent workflow to evaluate whether they reduce the interactions required to reach target valid cookie consent states while respecting legal and functional constraints.

The next section (Motivation) develops the practical reasons for the problem, presents evidence of user burden, and states the structural gap that motivates our approach.

1.1 Motivation

In recent years, the exponential growth of data collection on the web has significantly heightened concerns over user privacy. Websites widely employ cookies and tracking technologies to personalize content and services, but these processes often occur without transparent and informed user control. Regulatory frameworks such as the European Union's General Data Protection Regulation (GDPR) [1] and the ePrivacy Directive [2] have established stringent requirements for obtaining explicit and informed user consent prior to the collection and processing of personal data.

Despite these regulatory measures, implementing effective consent management mechanisms remains a significant challenge. Many existing cookie consent systems are either overly simplistic, failing to provide users with meaningful choices, or excessively complex, leading to user confusion, consent fatigue, and inadvertent agreement [3, 4]. Such shortcomings undermine both the legal validity of the consent obtained and the autonomy of users over their personal information.

Moreover, the complexity of privacy preferences and dependencies between different types of cookies and data processing activities further complicate the issue. Users often face cognitive overload when navigating lengthy consent dialogs with numerous options and unclear implications. This situation necessitates innovative approaches to streamline consent interactions while ensuring compliance and respecting user privacy.

The motivation behind this research stems from addressing these challenges by leveraging graph-theoretic models to represent and optimize user consent pathways in cookie-based systems. By modeling cookie consent decisions and their interdependencies as directed graphs, redundant or less critical decision paths can be identified and eliminated. This simplification aims to reduce the number of interaction steps for users and make the process more understandable and efficient, thereby enhancing the clarity and reliability of the consent obtained.

Building on Konstantinidis et al.'s [5] ideas about path elimination, this thesis moves the focus to the user interface. My motivation is to see whether a simple graph-based view of the UI can reveal practical changes that reduce effort for users while still respecting legal rules.

For example, consider a typical user visiting a digital-library website to download an audiobook chapter. The cookie banner displays several categories tailored to this context—such as Essential, Functional (player/download), Analytics, Personalization, Social Media, and Third-Party Services—across multiple panels. To review and make informed decisions, the user needs to click through each category, read unclear descriptions, and accept or reject each individually. Moving from the catalog to the book detail or download page can trigger the dialog again in a slightly different sequence. This results in more than ten interactions, often leading to confusion or blind acceptance of default options.

In contrast, a graph-based model represents these categories and their dependencies as a directed graph. Redundant or dependent categories can be collapsed, and essential consent paths can be highlighted. The user is then guided through a streamlined sequence focused on a few high-impact decisions, reducing the interactions required relative to the baseline flow while preserving the expressiveness and legal validity of the consent process.

1.2 Problem Statement

Web applications must obtain valid user consent before setting non-essential cookies. In practice, the consent journey is a workflow with several screens and clicks. On many sites this workflow becomes long, repetitive, or inconsistent, which makes it hard for users to reach a valid consent state with reasonable effort. Our aim is to simplify this workflow at the structural level so that users reach a valid consent state with less effort, without breaking any legal or technical rules.

Definitions

Valid consent state: A configuration that (i) supports granularity (fine-grained choices), (ii) is revocable at any time, and (iii) never disables essential cookies. The state must be stored and applied reliably by the system.

Structural view: We view the cookie consent journey as a sequence of UI states (pages/dialogs/tabs) connected by user actions (clicks, toggles, confirmations). This view lets us talk clearly about path length, repeated screens, and redundant branches.

Problem

The problem investigated in this thesis is how to simplify a baseline consent workflow while maintaining its accessibility and regulatory compliance.
The goal is to construct a simplified workflow in which all target consent states—such as essential-only,analytics-permitted, and consent withdrawn—remain reachable from the initial state. At the same time, the workflow should require fewer user interactions, including fewer clicks, screens, and redundant branches, compared to the baseline structure. Throughout the simplification process, all privacy-related constraints must be preserved, particularly those concerning granularity, revocability, and the treatment of essential cookies.
To achieve this, we apply graph-based edge-removal algorithms to simplify the consent structure by eliminating redundant or non-essential connections between nodes. Implicit dependencies between options (for example, activating Analytics that automatically enables provider-side scripts) are modeled as explicit constraints in the graph, making such relations transparent and auditable.

Success Measures

We evaluate success by whether users can reach their goal with fewer interactions, whether the flow avoids redundant branches that add effort without benefit, and whether every path still respects all privacy constraints so that no service is enabled without valid consent.

Scope

Our focus is the UI-layer consent workflow and its structural simplification. System-wide enforcement in other layers (client-side code, server-side processing, third-party services) is assumed and not audited here. Wording/visual design tweaks, behavioral user studies, and full legal audits are also out of scope for this section.

Research Question

How can graph-theoretic models be used to enforce user consent within cookie-based systems, while preserving system utility and adhering to privacy constraints?

1.3 Objectives

This section states clear, testable objectives aligned with the research question. The first objective is to map the baseline consent flow. We document the current Klaro-based flow by listing UI states (banner, “more options”, category tabs, toggles, confirma- tions) and user transitions; we also define the target valid consent states and the privacy constraints (granularity, revocability, essential cookies).

The second objective is to produce candidate simplified versions. We apply five edge-removal al- gorithms within the consent module to derive simpler alternative flows, while keeping all target states reachable and all constraints respected.

The third objective is to implement and measure. We deploy both the baseline and the simplified flows and record metrics.

Finally, the fourth objective is to evaluate and report honestly. We compare simplified flows against the baseline to evaluate whether user effort decreases; we check compliance with privacy constraints.

1.4 Thesis Roadmap

The remainder of this thesis is structured as follows. Chapter 2 provides the theoretical background, explaining how cookies and consent mechanisms work and how consent interactions can be represented as graphs. Chapter 3 reviews related research in consent management, privacy engineering, and graph-based optimization, identifying conceptual and methodological gaps. Chapter 4 presents the methodological setup, including the system requirements, stack configuration, and the architecture used to model cookie-consent interactions. Chapter 5 describes the implementation and application of five graph-reduction algorithms (Remove First Edge, Remove Random Edge, Brute Force, Remove Min-Cut, and Remove MinMC). It further presents the experimental results on the baseline graph (G0) and its optimized variants (G1 and G1b), followed by a discussion of what these results mean in practice. Chapter 6 elaborates on the findings, addressing limitations, threats to validity, and the broader implications for consent-system design. Finally, Chapter 7 concludes the thesis by summarizing the main contributions and providing an outlook on future research directions.

2 Background

The exponential growth of the internet and web-based services has dramatically increased the amount of personal data collected, processed, and stored by organizations worldwide. This digital transformation has brought about significant benefits in terms of convenience, personalization, and connectivity, but it has also introduced new risks and challenges related to user privacy [6]. The rise of data-driven business models, targeted advertising, and pervasive online tracking has heightened public awareness and concern about how personal data is used and protected.
This section introduces the formal and technical building blocks that underpin our approach to modeling and simplifying cookie-consent workflows in web applications. It provides concise definitions of five core concepts: cookies as technical tools for storing user data; consent as a legal and behavioral mechanism; Consent Management Platforms (CMPs), which present consent dialogs/banners and record user preferences; Content Management Systems (CMSs), which deliver site content and integrate a CMP; and graph-theoretic models as a formalism for representing user-interaction paths. Together, these concepts establish the theoretical and architectural basis for the graph-based methods explored in later chapters.

2.1 Cookies

Cookies are small text-based data fragments stored on a user's browser by web servers to maintain stateful information across sessions. Originally introduced in the mid-1990s for enabling shopping carts in early e-commerce platforms [7], cookies have evolved to support a wide range of web functionalities, including session tracking, user personalization, analytics, and advertising.

2.1.1 Structure and Classification of Cookies

Technically, a cookie consists of a name-value pair and optional attributes such as expiration time, domain, path, and security flags (e.g., `Secure`, `HttpOnly`). Cookies can be broadly classified as follows:

- **Session Cookies:** A session cookie is a temporary cookie that is erased when the user closes their web browser. They are commonly used to store information that is needed only for the duration of a browsing session, such as authentication status or temporary preferences [7].

```
// This cookie will be deleted when the browser is closed
document.cookie = "sessionId=abc123; path=/";
```

Listing 2.1: Example of a session cookie (no Expires/Max-Age) [8].

- **Persistent Cookies:** A persistent cookie (also called a "permanent cookie" or "stored cookie") is a cookie that is stored on a user's device for a set period of time, even after the browser is closed.
  Persistent cookies have an explicit expiration date or a maximum age attribute, and are used to remember information, preferences, or login status across sessions [7].

```
// This cookie will expire in 30 days
document.cookie = "userId=xyz789; path=/; expires="
  + new Date(Date.now() + 30*24*60*60*1000).toUTCString();
```

Listing 2.2: Example of a persistent cookie (expires in 30 days) [8].

- **First-Party Cookies:** First-party cookies are set by the domain that the user is currently visiting. In contrast, third-party cookies are set by domains other than the one shown in the address bar [9].

```
// On example.com
document.cookie = "userPref=darkmode; path=/";
```

Listing 2.3: Setting a first-party cookie on `example.com` with site-wide scope (`Path=/`).

  This cookie is only accessible to scripts and requests from `example.com`.

- **Third-Party Cookies:** Third-party cookies are cookies that are set by a domain other than the one the user is currently visiting. These cookies are typically set by external services (such as advertisers, analytics providers, or social media widgets) embedded on the website, and are used for tracking users across different sites [10].

```
// Loaded from a third-party domain in an iframe or script
document.cookie = "affiliateId=abc123; domain=tracker.com; path=/";
```

Listing 2.4: Setting a cookie for `tracker.com` (third-party context).

This cookie is only accessible to `tracker.com`, not `news.example.com`.

2.1.2 Cookie Mechanism and Usage

When a user visits a web page, the server may include a `Set-Cookie` header in the HTTP response. The browser stores this data locally and includes it in the `Cookie` header of subsequent requests to the same domain . For instance:

- `Set-Cookie: sessionID=abc123; Path=/; Secure; HttpOnly`
- `Cookie: sessionID=abc123`

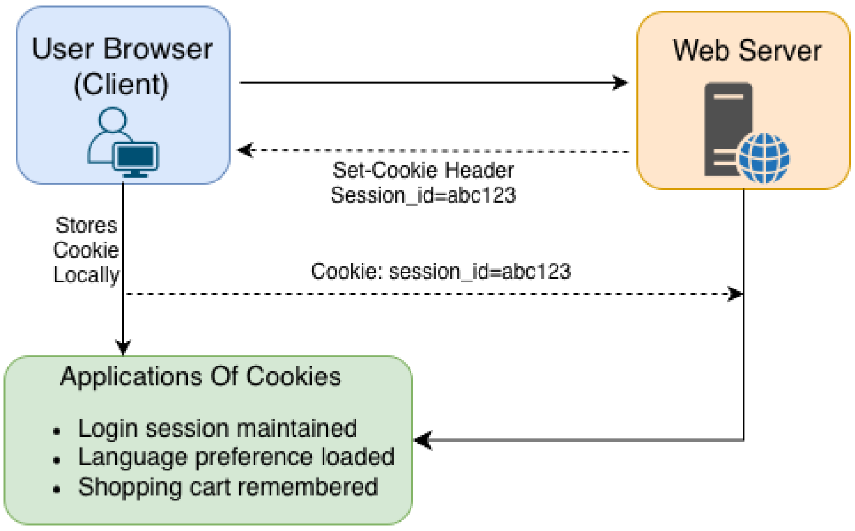


Figure 2.1: Workflow of cookie creation and usage between browser and server

This mechanism allows servers to identify returning users and maintain state, such as keeping a user logged in.

2.1.3 Applications of Cookies

Cookies play a crucial role in enabling a wide range of web functionalities that enhance both user experience and site operations. One of their primary uses is to maintain login sessions, allowing users to remain authenticated as they navigate between pages without repeatedly entering their credentials [7]. Additionally, cookies are commonly used to store user preferences, such as language selection and visual themes, ensuring that websites can deliver a personalized experience tailored to individual users [11]. In the context of e-commerce, cookies are essential for managing shopping carts, as they allow items selected by users to be retained across multiple browsing sessions and page visits. Beyond these convenience features, cookies also facilitate analytics and behavioral tracking, enabling website owners to gather data on user interactions and improve site performance. Furthermore, cookies underpin targeted advertising by allowing advertisers to deliver personalized ads based on users' browsing history and preferences [12].

2.1.4 Privacy Implications and Legal Frameworks

Cookies, especially third-party ones, raise significant privacy concerns due to their ability to track users across websites without informed consent [12]. In response, legal frameworks such as the **General Data Protection Regulation (GDPR)** [4] and

the **ePrivacy Directive** [13] require explicit user consent before setting non-essential cookies.
According to the GDPR, consent must be freely given, specific and informed, and unambiguous and affirmative [14].
Despite legal requirements, many websites employ deceptive design patterns (dark patterns) in cookie banners to manipulate user decisions [3].
This thesis treats "cookies" as the technical foundation of user tracking and preference storage in web applications. Since consent decisions on the web are ultimately implemented through these cookie mechanisms, understanding their structure and behavior is essential for designing privacy-compliant consent workflows. The optimization framework proposed in this work—aimed at reducing consent fatigue and simplifying user interactions—builds upon how cookies are created, categorized, and controlled across different layers of a consent workflow.

2.2 Consent

Consent is a fundamental principle in digital data governance, having evolved from a simple checkbox to a complex socio-technical construct. In the context of digital environments, consent is not only a legal requirement but also a multifaceted challenge involving legal, behavioral, and technical dimensions. According to Article 4(11) of the General Data Protection Regulation (GDPR), consent is defined as "any freely given, specific, informed and unambiguous indication of the data subject's wishes by which they, by a statement or by a clear affirmative action, signify agreement to the processing of personal data" [14]. This definition emphasizes four essential criteria: freedom of choice, specificity, informedness, and unambiguity. If these criteria are not met—such as when consent is bundled with mandatory services or when it is difficult to refuse—then the resulting agreement cannot be considered valid. The authenticity of consent is closely linked to the design of user interfaces. The GDPR requires that consent must be as easy to withdraw as it is to give, and that users must not suffer negative consequences for withdrawing consent. In practice, however, mechanisms for revocation are often hidden or unintuitive, making it challenging for users to exercise their rights. The European Data Protection Board (EDPB)[14] further clarifies that consent must be both demonstrable and reversible, requiring organizations to implement systems that allow for traceable records and accessible revocation options.
From a technical perspective, implementing effective consent mechanisms involves balancing legal obligations with usability and system efficiency. Privacy-by-design approaches advocate for embedding consent as a default feature, utilizing modular Consent Management Platforms (CMPs), consent receipts, and metadata tagging of user preferences. Despite these frameworks, challenges persist in ensuring transparency, synchronizing consent states across distributed systems, and reconciling usability with legal requirements. Even when compliant interfaces are deployed, users may still misunderstand what they are consenting to, highlighting a persistent gap between legal theory and practical user comprehension.
While much of the discussion around consent focuses on web applications, its significance is increasingly recognized in other digital domains, underscoring the need for ongoing research and innovation in consent management.

Beyond web applications, consent plays an increasingly critical role in a variety of domains, including health informatics, where informed consent is foundational for ethical medical research and patient data governance and regulates the collection, processing, and sharing of biometric, genetic, and clinical data [15]; artificial intelligence (AI), where consent governs the use of personal data to train models, particularly in systems involving user profiling or behavioral prediction [16]; mobile ecosystems, where smartphones and mobile apps depend on user consent to access sensitive features and where background data collection and third-party SDKs further complicate compliance [17]; smart homes and IoT, where devices such as voice assistants and wearables often rely on implicit or indirect forms of consent, requiring contextual and tiered consent frameworks [18]; and education technology, where platforms for minors require explicit parental consent and must comply with regulations such as the GDPR and the Children's Online Privacy Protection Act (COPPA)[19, 20].
These diverse and expanding applications demonstrate that consent is no longer limited to simple cookie pop-ups but has become a systemic requirement in the design of ethical, user-respecting digital infrastructure. It functions not only as a legal mechanism but also as a dynamic interface between users, technology, and society.
User "consent" is the legal and ethical cornerstone of this thesis. As modern privacy laws demand that consent be informed, explicit, and freely given, this research seeks to formally model the consent collection process. By representing consent as a series of user decisions encoded in graph structures, the goal is to analyze and optimize the paths users take when interacting with consent interfaces.

2.2.1 Consent Management Platforms (CMPs)

Consent Management Platforms are essential tools for implementing legal frameworks such as the General Data Protection Regulation (GDPR) and the ePrivacy Directive. Their core function is to serve as an interface between users and service providers, allowing for the collection, storage, and management of user consent regarding the processing of personal data. Contrary to popular belief, CMPs are not merely visual cookie banners; they consist of multiple technical components, including a user interface for displaying consent options, a backend for storing preferences, and APIs for communicating with third-party services.
CMPs typically present users with categorized information about different types of cookies, their processing purposes, and the third parties involved. These platforms often use multi-layered interfaces—such as expandable panels or "advanced settings" menus—enabling users to accept or reject data processing on a per-purpose or per-vendor basis. As such, the user experience (UX) design of CMPs plays a crucial role in shaping the authenticity of the consent provided.
Several studies, most notably those by Utz et al. (2019) and Nouwens et al. (2020), have shown that CMPs are frequently designed in ways that nudge users toward consent. The use of dark patterns—deceptive design strategies like visually dominant "Accept All" buttons and hidden or minimized "Reject" options—undermines the principle of freely given and informed consent required under Article 4(11) of the GDPR. As a result, while such interfaces may appear legally compliant, they fail to uphold the ethical and legal standards of meaningful user choice [4, 3].
Evaluating the legal legitimacy of CMPs reveals additional challenges. Many interfaces obscure or complicate the process of rejecting consent, placing the "Reject" button

behind multiple interaction layers, while highlighting the “Accept All” option with vivid colors and central placement. This imbalance violates the GDPR’s requirement for equally accessible choices and challenges the assumption that consent has been given voluntarily.
A closer inspection of popular CMPs such as Cookiebot, OneTrust, and Quantcast Choice demonstrates that their design decisions are often geared more toward achieving legal coverage than facilitating genuine user consent. For instance, Cookiebot tends to bury consent customization settings in nested menus, while prominently featuring an “Allow All” button. OneTrust, widely used by multinational corporations, allows for more granular settings but defaults to a design that streamlines the acceptance path. Quantcast Choice similarly provides detailed controls but uses button placement, color hierarchy, and interface compression to lead users toward consent [13].
Additional CMPs, such as TrustArc, Usercentrics, Didomi, and Civic, are also widely deployed across industries. While all claim GDPR compliance, their default configurations often exhibit similar usability issues. In free or pre-configured versions, dark patterns are frequently enabled by default, and modifying them requires technical expertise and backend access—resources not readily available to small websites or non-expert administrators [21].
In addition to these commercial CMPs, Klaro offers an open-source alternative focused on transparency and flexibility. Klaro allows full configuration through a lightweight JavaScript file and can be integrated directly into any website without external dependencies. Its open architecture makes it particularly suitable for research and experimental setups, as it enables developers to inspect, modify, and store user consent data in customized ways. In this thesis, Klaro serves as the consent management platform within the implemented system.
In conclusion, while CMPs play a vital role in ensuring legal compliance, they risk becoming instruments of legal façade if not implemented with fairness, transparency, and user-centered design. A legitimate CMP must offer users the ability not only to accept but also to reject and continuously manage their preferences in a clear and equal manner. Only then can consent be considered both legally valid and ethically sound.
Consent Management Platforms serve as operational bridges between legal consent requirements and technical implementation. In this thesis, CMPs are not only studied as interactive tools for users, but also structurally modeled as graphs to analyze how their design influences user behavior. The proposed approach applies graph path elimination algorithms to uncover and streamline consent journeys within the CMP interface.

2.2.2 Consent Management Systems (CMSs)

Consent Management Systems represent the back-end infrastructure necessary for organizations to ensure lawful, transparent, and user-respecting data processing in compliance with frameworks such as the General Data Protection Regulation (GDPR) and the ePrivacy Directive [1]. Unlike Consent Management Platforms, which primarily focus on the user-facing interface for collecting consent, CMSs offer the technical foundation for storing, auditing, updating, and revoking user consent throughout its lifecycle.
A typical consent management system comprises several tightly integrated modules

that manage the full lifecycle of user consent. A consent interface works with CMPs or internal UI components to capture users' data preferences, often using layered screens and purpose-based choices to enable granular consent [13]. The captured choices are then persisted by a storage engine that records consent metadata in a secure, timestamped, and immutable form, supporting privacy-compliant logging via secure databases and cryptographic techniques; for interoperability, the Kantara Initiative's Consent Receipt provides a structured format for storing and transmitting such data [22]. On top of this, an audit and compliance layer produces trails and reports for supervisory authorities and thereby supports GDPR Article 7(1), which requires controllers to demonstrate that valid consent was obtained [23]. Users can change their mind through a revocation and update service that enables real-time withdrawal and preference updates, aligning with GDPR's emphasis on the ease of withdrawing consent (Recital 42) [1]. Finally, an API gateway enforces consent and data-protection policies by mediating access to backend services and ensuring that external systems respect user preferences and regulatory requirements [24].
Distinction from Consent Management Platforms.these platforms are typically third-party or in-house tools that display consent banners and UI widgets to end users. Their scope is limited to presenting choices and capturing user input. However, a CMP without an underlying CMS cannot fulfill legal obligations, as it lacks the ability to store, validate, or revoke consent.
For instance, while platforms like OneTrust or Cookiebot offer CMP functionality, the full integration into a CMS enables the logging of consent in back-end databases, distribution of consent states across systems, and long-term compliance reporting [13]. Thus, CMPs may be seen as the interface layer, whereas CMSs are the operational backbone [21].

Data Storage, Processing, and Consent Lifecycle Management

A robust CMS ensures that user preferences are persistently stored in encrypted databases with versioning support, validated against processing policies before any action on personal data is taken, synchronized across distributed systems via event-driven architectures or microservice frameworks, and auditable so that organizations can produce detailed logs of when, how, and for what purpose consent was obtained, updated, or withdrawn.
For example, a CMS implementation might include a PostgreSQL or MongoDB database with schema tables tracking user ID, consent status per data purpose, consent source (CMP or manual API input), timestamps, and revocation events [21].

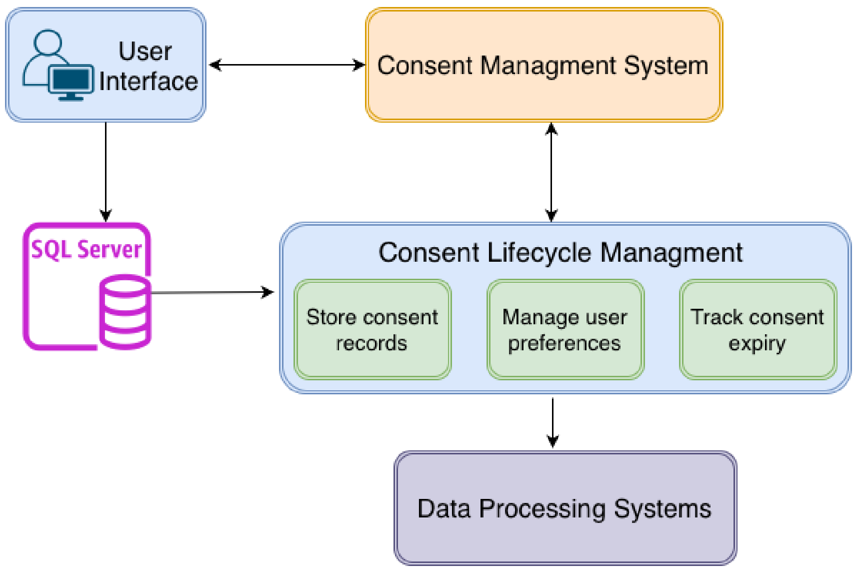


Figure 2.2: System architecture of a Consent Management System (CMS).

Figure Explanation:

Top Left: User Interface
The process begins with the User Interface (UI), where users interact with the system to set or modify their consent preferences.
Top Center: Consent Management System
The UI communicates directly with the Consent Management System (CMS), shown at the top center. This module acts as the central coordinator, receiving user input and orchestrating consent-related operations.
Center: Consent Lifecycle Management Below the CMS, the Consent Lifecycle Management module manages the entire lifecycle of user consent. This includes three primary functions:

- **Store Consent Records (Left):** Securely storing user consent records in the database.
- **Manage User Preferences (Center):** Handling ongoing user preferences related to data processing.
- **Track Consent Expiry (Right):** Monitoring consent expiration to ensure compliance and prompt renewals when necessary.

Bottom Left: Consent Database
The Consent Lifecycle Management module interfaces with the Consent Database (bottom left), where all consent records and related data are persistently stored.
Bottom Right: Data Processing Systems
On the bottom right, Data Processing Systems interact with the Consent Lifecycle Management module to validate user consent before any data processing activities occur.

2.2.3 Integration with Other Systems and Graph-Based Optimization

In advanced privacy-aware systems, such as those involving graph-based consent optimization, the CMS functions as the central registry of permissible user actions.

For instance, when a user logs into a digital library platform and selects privacy preferences via a CMP, the CMS stores the selection. These preferences are then converted into constraints within a directed graph representing possible interaction paths through the application.
Algorithms such as path pruning or edge removal can be applied to reduce redundant or unauthorized paths, ensuring that user flows respect privacy constraints. The CMS thus not only acts as a consent repository but also as a real-time decision engine for algorithmic enforcement.
As highlighted by Filipczuk et al. [5], the use of graph-theoretic models to represent consent workflows can significantly enhance transparency and accountability by embedding consent logic directly into the system's operational flow. The CMS plays a vital role here, feeding valid constraints and consent states into the optimization logic.
In the context of this thesis, the Consent Management System functions as the central infrastructure responsible for storing, updating, and managing user consent data in a structured and persistent way. Unlike traditional CMPs that rely on static, one-time interactions through cookie banners, CMS supports dynamic, ongoing consent tracking and integrates with graph-based models to analyze user navigation patterns. This dynamic architecture allows us to identify and optimize consent decision paths using elimination algorithms, reducing the number of required clicks while maintaining legal compliance and respecting user privacy.
As such, CMS is not only a technical foundation for the project but also a practical means to implement and evaluate user-centric privacy design.

2.3 Static vs. Dynamic Consent Management Systems

In the context of consent management in web applications, the terms "static" and "dynamic" refer to how systems process and respond to user choices over time.
Static system: Static consent management systems typically collect consent as a one-time event—often through a cookie banner—and do not modify the user interface or experience based on those choices. Once consent is recorded, the system remains unchanged, offering limited personalization or responsiveness. Many traditional Consent Management Platforms, such as default implementations of Cookiebot or Quantcast Choice, exemplify this static approach [13, 25].
By contrast, **Dynamic systems:** dynamic consent management systems continuously process user choices and adapt their behavior in real time. Such systems can leverage graph-based models to represent user decisions as weighted edges, enabling algorithmic optimization of consent paths and adaptive flows that enhance both user experience and regulatory compliance [26]. Dynamic systems not only track consent but also integrate it into interaction logic, allowing for ongoing updates and personalized consent journeys. This methodology, as proposed in this thesis, is inherently dynamic, as it continuously incorporates user preferences and optimizes decision paths accordingly.

2.4 Formal Definition and Role of Graphs in Cookie Consent Interaction Systems

Graphs are among the most widely used and fundamental mathematical structures in computer science and software engineering [27, 28]. They serve as an abstract model for representing binary relations between entities. In this thesis, directed graphs (digraphs) are employed to model user interactions with cookie consent systems, providing a rigorous foundation for analyzing and optimizing user paths [29].

2.4.1 Mathematical Definition of a Directed Graph

Formally, a directed graph G is defined as an ordered pair:

$$G = (V, E)$$

where: - V is a finite set of vertices (also called nodes), and - $E \subseteq V \times V$ is a set of directed edges. Each edge $(u, v) \in E$ represents a directed connection from vertex u to vertex v.
Note that in a directed graph, the presence of an edge (u, v) does not imply the presence of (v, u). Additionally, some graphs may contain weighted edges, where each edge has a numeric value or attribute associated with it [30].

2.4.2 Graph Properties

Graphs are versatile mathematical structures whose properties can be tailored to fit a wide range of application contexts in computer science and software engineering. The structural characteristics of a graph significantly influence its suitability for modeling different types of relationships and processes [30, 27].
A graph may be classified as simple if it contains no duplicate edges or self-loops, ensuring each connection between nodes is unique. The distinction between directed and undirected graphs depends on whether the edges have a direction; in directed graphs (digraphs), edges represent ordered pairs, while in undirected graphs, edges represent unordered pairs [30]. Weighted graphs assign numerical values to edges, representing quantities such as cost, probability, or importance, and are essential in applications like shortest path algorithms and network flow analysis [27].
Other important properties include whether a graph is cyclic or acyclic, indicating the presence or absence of cycles, and whether it is connected or disconnected, referring to the ability to reach all nodes from any starting node. These properties are fundamental when analyzing user navigation, network reliability, and decision-making processes [28].
In this thesis, we primarily use weighted directed graphs to represent user decision paths within a consent management workflow, as this structure allows for modeling and analysis of user interactions and system behavior.

2.4.3 Motivation for Graph-Based Modeling

Modeling user consent interactions as a graph provides a mathematically tractable representation of system elements and their relationships [30, 27]. This approach enables detailed analysis and optimization of user decision paths and allows us to apply well-established graph algorithms to enhance user experience while ensuring compliance with data-protection regulations [28]. Logical constraints and dependencies can be encoded as structural properties within the graph, offering a unified framework for both technical analysis and legal compliance (e.g., GDPR). Recent research shows that graph-based representations of consent workflows can improve system transparency and support formal reasoning about user interactions and data processing. Building on this foundation, we implement and evaluate five graph-based algorithms in this thesis: (A1) Remove First Edge, (A2) Remove Random Edge, (A3) Brute Force, (A4) Remove Min Cut, and (A5) Remove Min MC [5].

- **A1 — Remove First Edge [5].**

  The algorithm examines all paths that are reachable from a start node s to a target outcome t. For each such path, it selects the very first edge (the edge adjacent to s) and includes it in a cut set. After deduplicating this collection of "first edges", the algorithm removes them from the graph. This simple, deterministic baseline prunes as early as possible in order to observe the resulting effects on reachability.

  Algorithm 1: RemoveFirstEdge

  ```
  Input  : A graph $G$ and a set of constraints $\mathcal{N}$.
  Output : A graph $G$.
  forall $(s,t) \in \mathcal{N}$ do
      forall $path \in$ getAllEdgePaths($G, s, t$) do
          $e \leftarrow$ getFirstEdge($path$)
          if hasEdge ($G, e$) then
              updateDependencies($G, e$)
              removeEdge($G, e$)
          end
      end
  end
  ```

  Example. Suppose there are two parallel routes from s to t: $s \to a \to t$ and $s \to b \to t$. A1 selects the first edge of each route, namely (s,a) and (s,b). Removing these two edges disconnects s from t. This shows that concentrating cuts near the source can be quite aggressive.

- **A2 —Remove Random Edge [5].**

  The algorithm considers all edges that lie on currently reachable paths from the start node s to the target outcome t. From this set, it randomly selects one edge, removes it, and updates the graph accordingly. This process is repeated multiple times, producing a set of removed edges and the resulting pruned graph. Since edge selection is random, different runs of the algorithm may yield different outcomes.

Algorithm 2: RemoveRandomEdge

```
Input  : A graph $G$ and a set of constraints $\mathcal{N}$.
Output : A graph $G$.
forall $(s, t) \in \mathcal{N}$ do
    forall $p \in$ getAllEdgePaths($G, s, t$) do
        $edgeIndex \leftarrow$ getRandomInteger($1, |p|$)
        $e \leftarrow p[edgeIndex]$
        if hasEdge ($G, e$) then
            updateDependencies($G, e$)
            removeEdge($G, e$)
        end
    end
end
```

Example. Consider a tiny graph with edges $s \to a \to c \to t$ and a shortcut $s \to c$. Algorithm A2 first collects edges that lie on at least one $s \to t$ path and then removes one at random. If it removes the shortcut $s \to c$, only the longer route $s \to a \to c \to t$ remains (no disconnection, but no shortcut). If it removes $a \to c$, reachability from s to t is broken. This shows that A2 is a stochastic stress test: outcomes vary across runs and may either shorten detours or disconnect paths.

- **A3 —Brute Force [5].**

This approach exhaustively explores edge-removal subsets in increasing cardinality. For each candidate set of edges, the edges are removed and the required property is tested (e.g., that the start node s can still reach the target t for all mandated start→outcome pairs). The algorithm returns the smallest candidate that satisfies the constraint, or the best-found solution within a practical search cap when exhaustive enumeration is infeasible. Overall, the algorithm enumerates edge-removal candidates, evaluates reachability, and identifies the minimal pruning that maintains the required connectivity.

Algorithm 3: BruteForce

```
Input  : A graph $G = (V, E)$ and a set of constraints $\mathcal{N}$.
Output : A graph $G^*$.
$\mathcal{A} \leftarrow \varnothing$
forall $(s, t) \in \mathcal{N}$ do
    $\mathcal{A} \leftarrow \mathcal{A} \cup$ getAllEdgePaths($G, s, t$)
end
$multicuts \leftarrow$ cartesianProduct($\mathcal{A}$)
$maxUtility \leftarrow 0$
$G^* \leftarrow G$
forall $multicut \in multicuts$ do
    $G' \leftarrow G$
    $\pi', p \leftarrow \varnothing, \varnothing$
    forall $e \in E$ do
        $\pi'(e) \leftarrow \pi(e)$
        $p(e) \leftarrow \sum_{\mathrm{p} \in r(v)} w_p$
    end
    forall $e \in multicut$ do
        if hasEdge $(G', e)$ then
            updateDependencies($G', e$ ,$\pi', p$)
            removeEdge($G', e$)
        end
    end
    $utility \leftarrow$ U($G'$)
    if $utility > maxUtility$ then
        $maxUtility \leftarrow utility$
        $G^* \leftarrow G'$
    end
end
return $G^*$
```

Example. Consider two parallel routes from s to t: $s \to a \to t$ and $s \to b \to t$, plus a cross-edge $a \to b$. A3 checks all small removal sets. With single-edge removals, removing (s, a) keeps the route $s \to b \to t$ intact; removing (s, b) keeps $s \to a \to t$ intact; removing (a, b) changes nothing essential. Since either (s, a) or (s, b) alone removes the redundancy while preserving at least one $s \to t$ path, A3 returns a size-1 solution as the minimal pruning.

- **A4 — Remove Min Cuts [5].**

This algorithm models pruning as a cut problem between a start node s and a target node t. It computes a minimum cut, defined as the smallest set of edges whose removal disconnects s from t; in the weighted variant, the cut corresponds to the set of edges with the lowest total weight. Conceptually, the algorithm identifies the “cheapest bottleneck” that simultaneously blocks all $s \to t$ routes, rather than pruning edges individually along each path.

Algorithm 4: RemoveMinCuts

```
Input   : A graph $G = (V, E)$ and a set of constraints $\mathcal{N}$.
Output : A graph $G$.
$w \leftarrow \varnothing$
forall $e \in E$ do
    $w(e) \leftarrow \pi(e) \sum_{p \in r(v)} w_p$
end
forall $(s, t) \in \mathcal{N}$ do
    forall $e \in$ MinCut$(G, w, s, t)$ do
        if hasEdge $(G, e)$ then
            updateDependencies$(G, e)$
            removeEdge$(G, e)$
        end
    end
end
```

Example. Suppose there are two parallel routes from s to t: $s \to a \to t$ and $s \to b \to t$. A minimum cut of size 1 is enough: removing either (s, a) *or* (s, b) breaks all $s \to t$ paths. If edges carried weights (e.g., "cost of removal"), the algorithm would pick the cheaper of (s, a) and (s, b). This illustrates how min-cut targets a single bottleneck instead of pruning many edges along the routes.

- **A5 — Remove Min MC [5].**

  This algorithm generalizes the minimum-cut approach to handle multiple terminal pairs $P = \{(s_1, t_1), (s_2, t_2), \ldots\}$ that must be separated simultaneously. At each iteration, it examines all edges that still lie on at least one currently reachable $s_i \to t_i$ path, selects the edge with the highest aggregate impact (e.g., one that intersects many of these paths while preserving required connections), removes it, updates the graph, and repeats this process until all specified pairs are disconnected or a defined removal budget is reached. The output is the set of removed edges that collectively sever the targeted connections across multiple pairs, serving as a greedy approximation to the multicut problem.

Algorithm 5: RemoveMinMC

```
Input   : A graph $G = (V, E)$, a set of constraints $\mathcal{N}$.
Output  : A graph $G$.
$w \leftarrow \varnothing$
forall $e \in E$ do
    $w(e) \leftarrow \pi(e) \sum_{p \in r(v)} w_p$
end
$multicut \leftarrow$ MinMC($G, \mathcal{N}, w$)
forall $e \in multicut$ do
    if hasEdge $(G, e)$ then
        updateDependencies($G, e$)
        removeEdge($G, e$)
    end
end
```

Example. Suppose we need to separate two pairs, (s,t) and (s,u). There are two routes to each target: $s \to a \to t$ and $s \to b \to t$, and $s \to a \to u$ and $s \to c \to u$. The edge $s \to a$ lies on both $s \to t$ and $s \to u$ paths, so the first greedy step removes (s,a) because it hits two pairs at once. After that, only $s \to b \to t$ and $s \to c \to u$ remain, and the two pairs are already separated. This shows how the greedy multicut prefers shared upstream edges that eliminate several overlaps with a single removal.

Each algorithm applies a distinct strategy to optimize or constrain user paths through the consent interface, as discussed in detail in subsequent chapters. Graph theory thus offers a formal and flexible framework for modeling consent flows, where each node represents a user state and each edge corresponds to a specific action or choice. By leveraging graph-based algorithms such as path elimination and cut reduction, this thesis aims to streamline user interactions, reduce cognitive load, and enhance fairness and usability in consent management systems.

Scope. Our focus is the UI-layer consent flow and its structural properties. System-wide enforcement beyond the UI is assumed and not audited here; analysis of cross-site tracking is out of scope.

Bridge to the model. Building on these definitions, Chapter 4 formalizes the consent journey as a directed graph, states the simplification problem under privacy constraints, and introduces the algorithms, instrumentation, and evaluation protocol used in this thesis.

3 Related Work

The management of user consent for cookies and personal data has emerged as a central challenge in the design of modern web systems. As the internet has evolved from a static information repository to a dynamic, interactive platform, the collection and processing of user data have become ubiquitous. Early web applications provided little transparency or control over data collection, and user consent was often implicit or overlooked entirely [7].
The proliferation of cookies and third-party tracking technologies in the late 1990s and early 2000s highlighted the need for robust mechanisms to inform users and obtain their consent [10, 12]. This historical context set the stage for a series of regulatory and technical developments that continue to shape the landscape of consent management.
This review proceeds from legal and empirical foundations to technical tracking ecosystems, then to graph-theoretic models, culminating in algorithmic approaches that inform our contribution.

The introduction of comprehensive legal frameworks, most notably the General Data Protection Regulation (GDPR)[14] and the ePrivacy Directive[31], marked a watershed moment for user privacy and consent. These regulations established explicit requirements for informed, specific, and freely given consent prior to data collection or processing. They also mandated transparency, data minimization, and the right to withdraw consent at any time. The impact of these frameworks has been profound, forcing organizations worldwide to rethink their data practices and invest in compliance infrastructure.
Empirical studies have shown that many websites implement consent mechanisms that technically comply with the law but fail to empower users. Degeling et al.[13] found that cookie banners often use pre-selected options, ambiguous language, and interface nudges to steer users toward acceptance. Nouwens et al.[25] and Utz et al. [4] demonstrated that interface design—such as the placement and prominence of "Accept All" versus "Reject All" buttons—can significantly influence user choices, often at the expense of genuine consent.
Representative studies report measurable shifts in user choices under interface changes (e.g., button salience, default states) and increases in erroneous or uninformed consent under complex option layouts, underscoring the need for structural remedies.

In response to regulatory requirements, Consent Management Platforms (CMPs) have become ubiquitous. Commercial solutions like Cookiebot, OneTrust, and Usercentrics, as well as open-source tools such as Klaro! [32], offer configurable banners, compliance dashboards, and consent logging. These platforms have enabled organizations to demonstrate compliance and manage user preferences at scale.
Yet, a critical review of related work must go beyond surface features to examine usability and user experience. Habib et al.[33] found that many CMPs present users with complex, overwhelming interfaces that lead to "consent fatigue." Matte et al.[34] and Santos et al.[35] document systematic violations of GDPR, such as hidden opt-out buttons and confusing defaults.Bielova et al. [36] show that manipulative design patterns—"dark patterns"—are still common, resulting in users granting broad consent with little reflection. These findings highlight a recurring theme: CMPs often prioritize organizational compliance over meaningful user control.
Unlike CMPs that prioritize auditability, logging, and preference storage, prior work rarely optimizes the user's decision path through banners and settings. Our focus diverges from compliance dashboards toward algorithmic restructuring of the consent journey, reducing effort while preserving transparency.

Beyond the user interface, the technical landscape of web tracking is increasingly complex. Englehardt and Narayanan [12] conducted a large-scale study of one million websites, revealing a dense ecosystem of third-party trackers. Acar et al. [37] showed that fingerprinting and other persistent tracking techniques can circumvent traditional consent mechanisms, further undermining user autonomy. These studies underscore that consent is not just a matter of interface design but also a structural and technical challenge.
Given the scale and evasiveness of tracking practices, structural models are needed to reason about permissible flows and user-facing constraints beyond interface tweaks.
Despite regulatory and technical advances, few studies propose algorithmic frameworks for simplifying user interaction and consent. The gap here is clear: while much research highlights the risks and shortcomings of current consent mechanisms, there is little work on formal, algorithmic approaches to optimize user decision paths.

Graph theory offers a powerful abstraction for modeling relationships, flows, and dependencies in complex systems. It has been widely applied in privacy engineering, from analyzing social networks [38] to modeling privacy-preserving data flows [39] and evaluating re-identification risks [40]. These applications demonstrate how graph structures can reveal vulnerabilities, support constraint enforcement, and enable formal reasoning.
However, most graph-based privacy research has focused on backend processes—such as auditing, anonymization, or access control—rather than the front-end user experience. Munir et al. [41] developed CookieGraph to detect and classify cookie synchronization, addressing surveillance practices but not the usability of consent processes. The opportunity, therefore, lies in reframing consent interactions themselves as graph optimization problems, enabling algorithmic simplification and improved user experience.

Algorithmic Approaches to Consent Flow Optimization relevant work is by Konstantinidis et al. [5], who formalized consent management as a graph problem through the Consented Data Workflow (CDW) model. In this framework, data processing pipelines are represented as directed acyclic graphs (DAGs), with edges corresponding to possible data flows and user consent imposing constraints on permissible paths. They introduced a path-elimination algorithms—RemoveMinCuts, RemoveRandomEdge, RemoveFirstEdge, RemoveMinMC, and BruteForce—each offering different trade-offs between scalability and optimality. While brute force guarantees minimal solutions, it is computationally intractable for large graphs; heuristic methods scale better but may yield suboptimal results. Their work provides a rigorous theoretical foundation but remains largely simulation-based, leaving open questions about real-world performance and usability.
Where Konstantinidis et al. evaluate path elimination on simulated DAGs, we operationalize these algorithms within an operational web stack (CMS + CMP) and evaluate user-centric metrics (clicks to resolution, time-to-consent, error rate) under interface constraints and legal checks.
Parallel research has explored alternative models for structuring privacy decisions. Early standards like P3P [42] and EPAL [43] focused on machine-readable policies but neglected user interaction. Later, structured decision-dialog frameworks sought to guide users step-by-step through privacy decisions [44], improving comprehension but lacking adaptability. In contrast, graph models support dynamic restructuring of user paths, enabling the pruning of redundant branches in ways that decision trees cannot. Work on knowledge graphs and recommender systems [45] shows that eliminating low-utility paths can reduce cognitive load and improve satisfaction, reinforcing the potential of graph-based path elimination for consent systems.
Decision-tree dialogs can improve comprehension but remain largely static; dynamic-consent systems increase flexibility yet demand heavy infrastructure and often user profiling. Graph-based elimination offers a lightweight, transparent middle ground by pruning interaction paths without profiling, preserving privacy while improving efficiency [46].
Dynamic consent has emerged as a response to the limitations of static, one-time consent models. Originally developed in healthcare and genomics, platforms like CTRL [47] and ATHENA [48] allow participants to revisit and adjust their consent over time, emphasizing flexibility and transparency. While these systems demonstrate the value of iterative engagement, they often require heavy infrastructure and are limited to specific domains. To translate these ideas to the web, lightweight dynamic models are needed that restructure interactions without overwhelming users. Studies on adaptive user interfaces suggest that systems that adjust to user behavior can improve trust and usability, but typically rely on machine learning and profiling. In contrast, graph-based elimination provides a transparent, lightweight mechanism for adaptivity: by pruning unlikely or redundant paths, the system can simulate adaptivity without deep behavioral profiling, preserving privacy while enhancing usability [49].

Research on interface design and dark patterns further underscores the need for structural optimization and showed how subtle design choices can manipulate perceptions of autonomy [50], while Dincelli et al. [51] warned that nudging strategies, though effective, raise ethical concerns. Bielova et al. [36] empirically demonstrated

that CMPs continue to exploit deceptive patterns, undermining the legitimacy of consent. These findings highlight the importance of moving beyond interface tweaks to algorithmic restructuring of consent flows. Graph-based methods align with ethical design principles by reducing complexity without resorting to manipulation.
In summary, regulatory and empirical studies that expose usability shortcomings in consent mechanisms, CMP implementations that emphasize compliance but neglect workflow efficiency and graph-theoretic approaches that provide strong theoretical insights but remain untested in operational web contexts. Additional contributions from dynamic consent models, decision tree interfaces, recommender systems, and adaptive UIs highlight the breadth of strategies for improving user decision-making, but none directly address the optimization of cookie consent paths at the interaction layer. The gap, therefore, lies in the absence of approaches that algorithmically restructure consent workflows, simultaneously minimizing user effort, preserving transparency, and ensuring legal compliance.
We define optimization targets as (i) reducing the number of clicks needed to reach a valid consent state, (ii) shortening the path a user must follow when making consent decisions, and (iii) preserving user control and transparency during the graph simplification process. legality is preserved by explicit constraints ensuring required notices and granular choices remain available.
Accordingly, our contribution is the first end-to-end operationalization of graph-based path elimination for cookie consent in an operational CMS,+,CMP stack, demonstrating measurable reductions in user effort while maintaining legal and ethical constraints.

3.1 State of the Art in Consent Management and Privacy Engineering

The management of user consent for cookies and personal data has become a critical focus in web development and privacy engineering. Regulatory frameworks such as the General Data Protection Regulation (GDPR) and the ePrivacy Directive [14, 31] mandate explicit, informed consent for data collection, reshaping requirements for web applications.
To operationalize these requirements at scale, a broad ecosystem of Consent Management Platforms (CMPs)—both commercial and open-source—provides configurable banners, compliance dashboards, and consent logging. These systems enable organizations to manage preferences and demonstrate legal compliance, yet their design choices strongly shape user outcomes.
Recent empirical work highlights significant shortcomings in prevailing consent mechanisms. Studies show that many CMP implementations employ manipulative or confusing interfaces—e.g., pre-selected options, asymmetric salience of "Accept All" versus "Reject All," or hidden opt-outs—that nudge users toward broad consent and erode autonomy [13, 25, 4]. Further evidence documents "consent fatigue," where complex, repetitive dialogs overwhelm users and degrade decision quality [33, 34].
Technically, consent management has evolved into a complex, multi-layered infrastructure, involving client-side scripts, third-party services, and backend databases that must remain synchronized to reflect user preferences accurately. Large-scale measurements reveal dense third-party ecosystems and complex dependencies between

services, showing that user choices made through consent dialogs often have indirect effects that are difficult to trace or control.
In response, researchers have explored formal and algorithmic approaches. Graph-theoretic formulations model interactions, data flows, and consent pathways, enabling analysis and optimization over constrained decision spaces [5]. For example, the Consented Data Workflow (CDW) frames consent as constraints over directed acyclic graphs, which supports path-elimination strategies aimed at reducing user effort while preserving legality.
Taken together, the state of the art provides (i) robust legal foundations and compliance tooling, (ii) convergent empirical evidence on usability pitfalls and dark patterns, and (iii) formal graph-based tools for constrained optimization. What remains under-explored is an operational evaluation of path-elimination within real web stacks that preserves legal constraints while improving user-centric metrics (e.g., clicks to valid consent and time). This motivates the synthesis and gap analysis that follows and sets the stage for our system model and evaluation.

3.2 Synthesis, Gap, and Positioning

Synthesis. The reviewed literature highlights significant progress in the development of consent management systems, regulatory frameworks, and user interface design for privacy and consent. Legal mandates such as the GDPR and ePrivacy Directive have established foundational requirements for explicit, informed consent, prompting the widespread adoption of Consent Management Platforms (CMPs) across the web. Empirical studies, however, consistently reveal that many CMPs prioritize compliance over usability, often employing interface designs that nudge users toward broad consent and contribute to consent fatigue [13, 25, 33].
From a technical perspective, modern consent frameworks operate within increasingly complex ecosystems of third-party services and embedded scripts, which can reduce the transparency and reliability of traditional consent banners [12, 37]. While some solutions focus on backend privacy enforcement, few address the optimization of user-facing consent flows. Recent advances in graph-based modeling offer promising formal frameworks for representing and analyzing consent interactions, but existing work has largely remained theoretical or simulation-based [5].

Gap. Despite these advances, a clear research gap persists. Current systems rarely integrate algorithmic optimization with user-centric design to streamline consent interactions, minimize user effort, and enhance transparency. Most notably, there is a lack of practical implementations that leverage graph-based algorithms to restructure consent workflows at the interaction layer, ensuring both legal compliance and improved user experience.

Positioning. This thesis positions itself at this intersection, it operationalizes graph elimination algorithms within a real-world content management environment, bridging the gap between theoretical models and practical deployment. By focusing on the optimization of consent paths in user interfaces, this work aims to advance the state of the art in privacy engineering, offering a novel, adaptive, and user-friendly approach to consent management.

4 Methodology

This chapter explains how we modelled the website-wide cookie–consent journey as a directed, labelled graph; how we designed two streamlined variants of the experience (G1 and G1b); how we implemented the toolchain; and how we measured user effort in terms of clicks. All numerical outcomes, tables, and figures produced by this pipeline are reported in Chapter 5.

4.1 Specifications and Requirements

Goal. Reduce the number of user interactions required to reach target outcomes (e.g., secured login, media playback, payments) *without* weakening privacy-by-default. Non-essential services remain OFF until the user consents; consent must be explicit, traceable, and reversible.

Scope and completeness. The model covers the Consent Management UI (banner+modal), all site sections that can surface consent prompts, and all non-essential services that are gated by consent. The graph captures multiple legitimate routes (e.g., first-time banner, "manage cookies", granular toggles, accept-all) because different users may take different paths.

Automation and reproducibility. Algorithms operate on the *entire* graph—no manual, hand-picked edge deletions.

What we measure

For every `start` node (entry points such as `Entry`, `BannerToast`, `Home`, `Discover`, `SearchBooks`, `Audiobooks`, `DigitalMagazine`, `Membership`, `ContactUs`, `Login`, and also `Manage`, `Modal`, `LetMeChoose`, `AcceptAll`, `Save`) and for each `outcome` (`Out_BasicAccess`, `Out_SecuredLogin`, `Out_AllMedia`, `Out_Payments`, `Out_Personalized`, `Out_AllContent`, `ConsentSaved` we compute two costs:

- **Consent clicks** — the number of explicit consent actions on the path: `click(accept_all)`, `click(save)`, any granular `toggle ON`, and (in G1b) `click(auto_apply)`. Edges labelled `implicit_first_use` cost zero in this metric.

- **Total clicks** — all UI interactions on the path that require a user click (including `click(manage)`, opening the modal, etc.). Pure dependencies like `requires(...)` carry zero cost here.

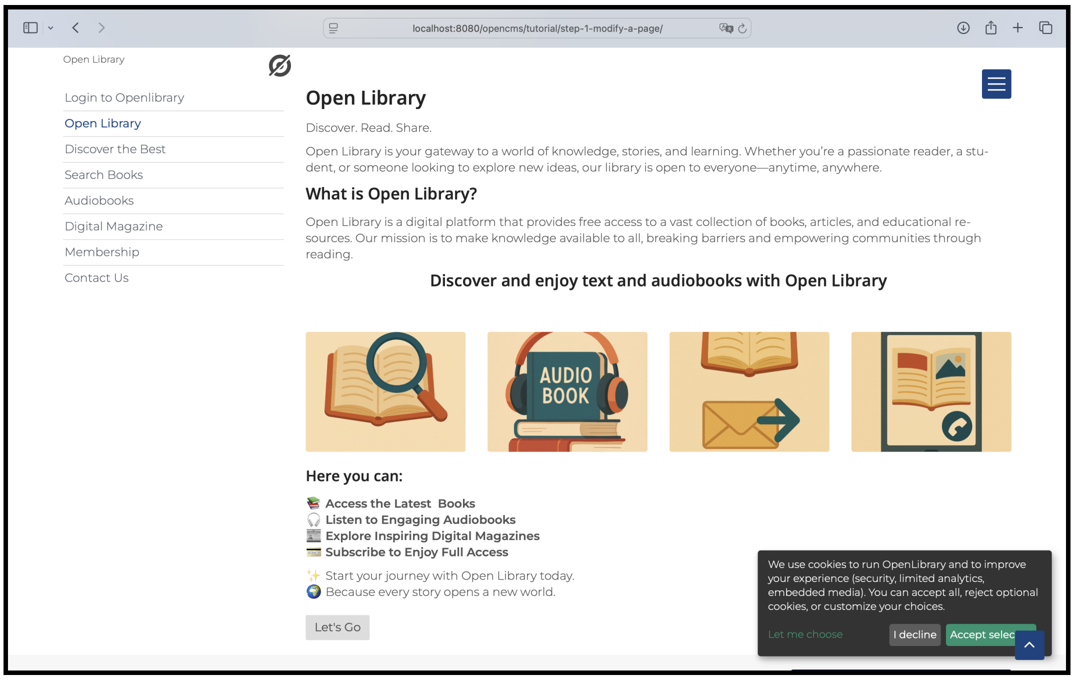
Costs are defined per edge via labels; minimum-cost paths are computed with Dijkstra on the labelled directed graph. We compare costs *before* and *after* an algorithmic change or a design variant.

4.2 System Stack and Setup

We implemented the demo website locally. The stack is: Figures 4.1–4.2 show the cookie banner, the Klaro modal with toggles.

Table 4.1: Local system stack and setup with consent storage.

Component	**Technology**	**Details / Purpose**
CMS / Rendering	OpenCms (Tomcat views)	Renders pages: Home, Login, Discover, Search, Audiobooks, Digital Magazine, Membership, Contact.
Server Runtime	MAMP (Apache, PHP)	Hosts PHP endpoints for server-side actions.
Endpoints	PHP	Book Search, Audiobook Search, Contact Form, Membership.
Database	MySQL (`opencms`)	Stores shared data: Search metadata, subscriptions, contact tickets, consent preferences.
Consent Mgmt	Klaro CMP	Manages cookie consent with categories and sub-cookies.
Data Storage	Cookie & MySQL	CMP consent cookies are stored on localhost and in the database. User decisions are also saved in MySQL: – If the user is registered, consents are linked to the username. – If the user is not registered, consents are stored under a guest profile.



Open Library

Discover. Read. Share.

Open Library is your gateway to a world of knowledge, stories, and learning. Whether you're a passionate reader, a student, or someone looking to explore new ideas, our library is open to everyone—anytime, anywhere.

What is Open Library?

Open Library is a digital platform that provides free access to a vast collection of books, articles, and educational resources. Our mission is to make knowledge available to all, breaking barriers and empowering communities through reading.

Discover and enjoy text and audiobooks with Open Library

Here you can:

📚 Access the Latest Books
🎧 Listen to Engaging Audiobooks
📰 Explore Inspiring Digital Magazines
💳 Subscribe to Enjoy Full Access

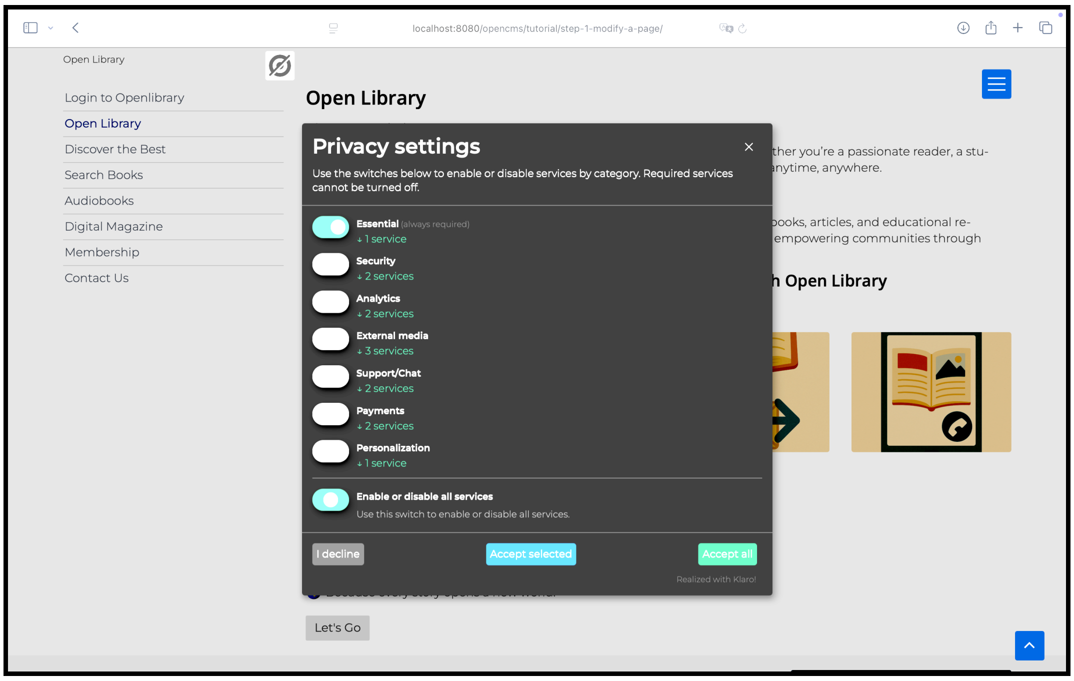
✨ Start your journey with Open Library today.
🌍 Because every story opens a new world.

We use cookies to run OpenLibrary and to improve your experience (security, limited analytics, embedded media). You can accept all, reject optional cookies, or customize your choices.

Figure 4.1: Cookie banner on the home page (Klaro).

Table 4.2: Klaro CMP categories and mapped services.

Category	**Example Services**	**Notes**
Essential (always on)	Core cookies	Non-optional functionality.
Security	Google reCAPTCHA; Cloudflare Turnstile	Bot and abuse protection.
Analytics	Google Analytics 4 (GA4); Matomo	Usage statistics and analysis.
External media	YouTube; Vimeo; Google Maps	Embedded third-party content.
Support / Chat	Tawk.to; Intercom	Live chat / helpdesk widgets.
Payments	Stripe; PayPal	Payment processing.
Personalization	On-site recommendations	Content/UI personalization.



localhost:8080/opencms/tutorial/step-1-modify-a-page/

Open Library

- Login to Openlibrary
- Open Library
- Discover the Best
- Search Books
- Audiobooks
- Digital Magazine
- Membership
- Contact Us

Open Library

Privacy settings

Use the switches below to enable or disable services by category. Required services cannot be turned off.

- [✓] **Essential** (always required) ↓ 1 service
- [ ] **Security** ↓ 2 services
- [ ] **Analytics** ↓ 2 services
- [ ] **External media** ↓ 3 services
- [ ] **Support/Chat** ↓ 2 services
- [ ] **Payments** ↓ 2 services
- [ ] **Personalization** ↓ 1 service

[✓] **Enable or disable all services**
Use this switch to enable or disable all services.

I decline | Accept selected | Accept all

Realized with Klaro!

...ther you're a passionate reader, a stu-
...nytime, anywhere.

...books, articles, and educational re-
...empowering communities through

...h Open Library

Let's Go

Figure 4.2: Privacy settings (Klaro banner) with categories.

4.3 Architecture and System Design

The baseline graph (G0)

We model the consent journey in Graphviz DOT (`artefacts/graphs/G0.dot`) with three layers:

1. **UI/CMP layer** — `Entry`, `BannerToast`, `LetMeChoose`, `Modal`, `Manage`, actions `AcceptAll`, `AcceptSelected`/`Save`, `ToggleAll`, granular toggles `AccSecurity`, `AccAnalytics`, `AccExtMedia`, `AccSupport`, `AccPayments`, `AccPersonal`.
2. **Pages layer** — `Home`, `Discover`, `SearchBooks`, `Audiobooks`, `DigitalMagazine`, `Membership`, `ContactUs`, `Login`, each with realistic navigation and a `click(manage)` path into the CMP.
3. **Services layer (gated)** — `SecuritySvc`, `AnalyticsSvc`, `ExtMediaSvc`, `SupportSvc`, `PaymentsSvc`, `PersonalSvc`, plus `Essentials`.

Pages link to services via `requires(...)` edges; UI actions "unlock" services via `AcceptAll`/`Save`/`Acc*` $\rightarrow$ `Service`. Outcomes include `Out_BasicAccess`, `Out_SecuredLogin`, `Out_AllMedia`, `Out_Payments`, `Out_Personalized`, `Out_AllContent`, `ConsentSaved`.
Sanity check. The full G0 contains 39 nodes and 81 edges (validated with `pydot/networkx`). For some analytics we also use a filtered view (73 edges) that omits non-participating edges; both are reported in the Results section for transparency.

Baseline consent graph (G0)

The graph **G0** models the status quo of the consent workflow on the Open Library site. Its purpose is to capture, in a single directed graph, the consent UI (CMP),

the site pages where a journey can start, the non-essential services that are gated by consent (purposes), and the observable outcomes after consent has been applied. This baseline contains **39 nodes** and **81 directed edges** (unique), laid out left-to-right for readability. Importantly, G0 contains no quick shortcuts and no automatic persistence of consent; every consent change must be committed explicitly by the user.
Conceptually, the nodes fall into four families that interact tightly. First, the UI family represents the Klaro banner and modal: `Entry` (first visit) leads to `BannerToast`; users can `LetMeChoose` to open the `Modal`, or press `AcceptAll` or `AcceptSelected`, or `Decline`. Inside the modal there is `ToggleAll`, the commit button `Save`, and six per-purpose toggles: `AccSecurity`, `AccAnalytics`, `AccExtMedia`, `AccSupport`, `AccPayments`, and `AccPersonal`. These `Acc*` nodes change the selection only; in G0 they do not take effect until the user presses `Save`. The unique terminal state of the UI is `ConsentSaved`, denoting that the current selection has been persisted (to localhost and database). The only two ways to reach `ConsentSaved` are: (i) pressing `AcceptAll` on the first layer, which unlocks all non-essential purposes in one step, or (ii) choosing individual purposes in `Modal` and then pressing `Save`.
This interaction flow of the Klaro banner and modal is illustrated in Figure 4.3.

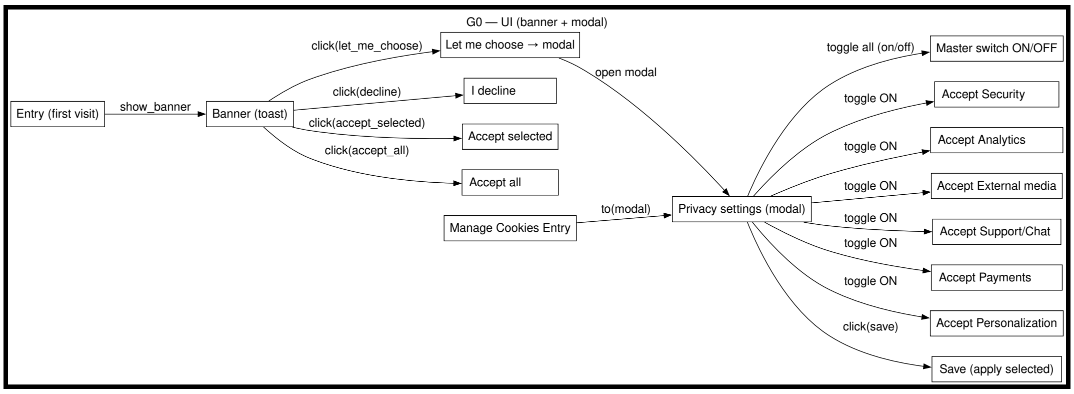


Figure 4.3: G0 — UI/CMP layer: User interactions from banner and modal to consent saving.

Second, the pages family models typical entry points of a user journey: `Home`, `Discover`, `SearchBooks`, `Audiobooks`, `DigitalMag`, `Membership`, `ContactUs`, and `Login`. Each page links to `Manage` (ManageCookies entry), which brings users back to the modal when they want to adjust their choices. Sample navigation edges between pages (e.g., `Home` → `Discover`) make explicit that consent interaction happens in the context of real browsing.
The navigation structure and the ManageCookies entry points are shown in Figure 4.4.

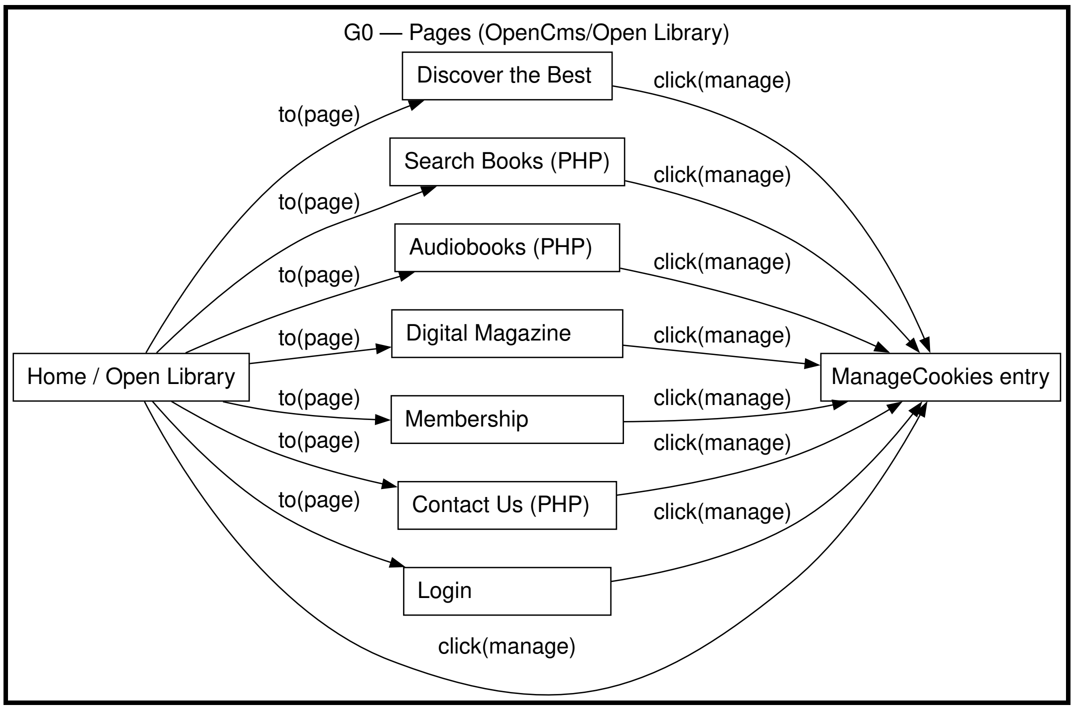


Figure 4.4: G0 — Pages layer: Navigation paths and ManageCookies entry points in Open Library.

Third, the services family contains non-essential capabilities that are gated by consent: `AnalyticsSvc`, `ExtMediaSvc`, `SupportSvc`, `PaymentsSvc`, `PersonalSvc`, and `SecuritySvc` (with `Essentials` always on). Edges of the form `requires(purpose)` from pages to services record which purpose a page depends on (e.g.,`DigitalMag` → `requires(analytics)`, `SearchBooks` → `requires(external_media)`).
Conversely, edges from `Save` to each service (`selected?  unlock`) and direct edges from `AcceptAll` to all services encode when a service becomes unlocked. This distinction—“needs” (dependency) versus “unlocks” (after consent)—is the backbone of path analysis: if a page requires a service but the service is not yet unlocked, the user must detour into the consent UI.
The page–service dependencies and unlock semantics are visualized in Figure 4.5.

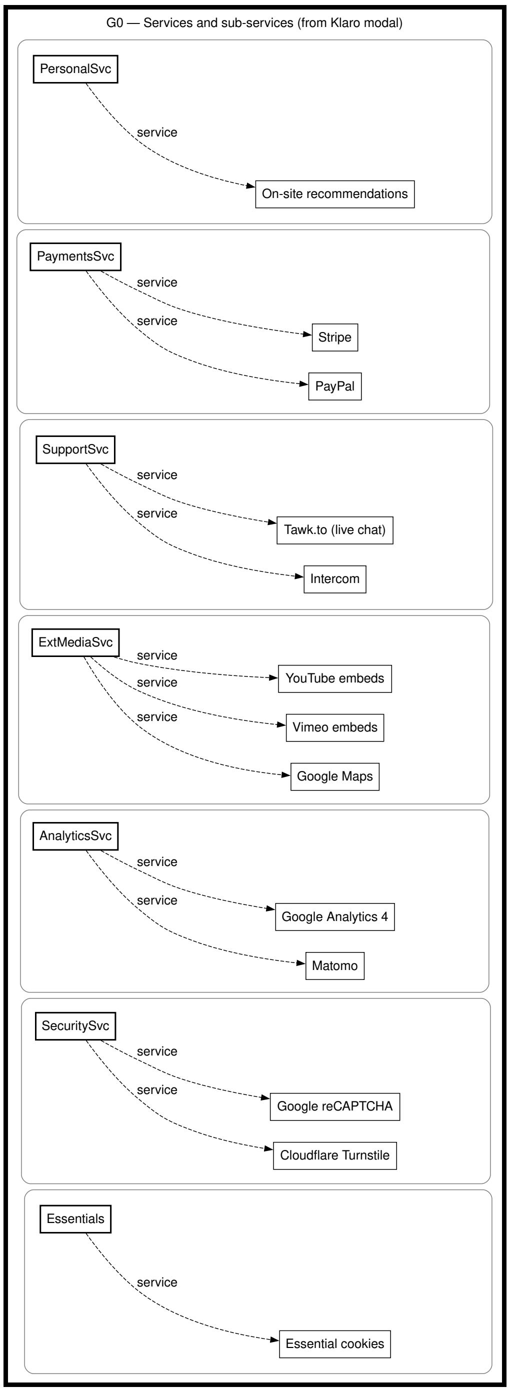


Figure 4.5: G0 — Services layer: Consent-gated purposes and sub-services.

Fourth, the outcomes family captures observable post-consent capabilities, such as `Out_BasicAccess`, `Out_AllContent`, `Out_AllMedia`, `Out_Payments`, `Out_Personalized`,and `Out_SecuredLogin`. Reaching one of these nodes in measurement indicates that the user has achieved the intended capability (e.g., playback of external media or activation of analytics).
The resulting observable post-consent capabilities are summarized in Figure 4.6.

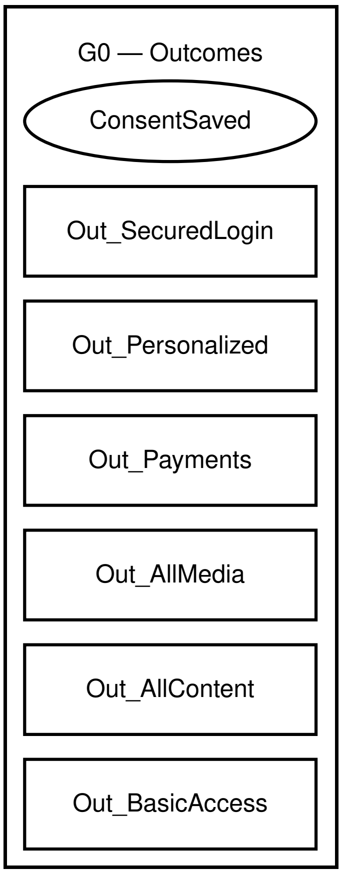


Figure 4.6: G0 — Outcomes layer: Observable capabilities after consent.

Edge labels in G0 serve three roles. (i) UI interactions such as `click(accept_all)`, `click(save)`, `open modal`, and `toggle ON` describe actual interface actions. (ii) Dependencies `requires(...)` from pages to services state prerequisites. (iii) Unlock semantics express when services become available: `selected? unlock` from `Save` (conditional on prior selections) and unconditional unlocks from `AcceptAll`. In addition, certain node labels include small markers like "(+2)" or "(+3)"; these encode the consent-portion click cost used in our path-length accounting for the results tables.

Behaviorally, G0 is intentionally conservative: there is no auto-apply. Switching `Acc*` toggles only prepares a selection; nothing is persisted until `Save` is pressed, and therefore consent clicks are relatively high for most content starts (typically two clicks: enter the consent UI and save). The only true one-step path is `AcceptAll`, which unlocks everything immediately and leads straight to `ConsentSaved`; while this shortens many paths, it may not reflect granular user preferences and is treated distinctly in our analysis.

Two examples illustrate the mechanics. If a user starts on `Home` and wants to view an embedded preview that depends on external media, the edge `Home` $\rightarrow$ `requires(external_media)` is active, but `ExtMediaSvc` remains locked until the user detours to `BannerToast`/`Modal`, turns on `AccExtMedia`, and presses `Save`. Only then can the path proceed to the corresponding outcome. By contrast, pressing `AcceptAll` at entry unlocks all services and yields immediate access to outcomes that would otherwise require multiple steps.

Finally, G0 serves as the control in our evaluation. For every start–outcome pair we compute the shortest path length L_{G0} and, where relevant, the shortest path to `ConsentSaved`. All deltas in the results section are defined relative to this baseline (e.g., $\Delta = L_{\mathrm{G1}} - L_{\mathrm{G0}}$). Because G0 requires an explicit `Save` (or `AcceptAll`) to commit consent, any reductions in consent clicks or total path length observed in later graphs can be attributed unambiguously to the UI/semantic changes introduced in G1 and G1b. In short, G0 is a faithful picture of a classic CMP flow—banner → modal → per-purpose selection → save—and a reliable baseline against which improvements are measured.

Design variants G1 (Quick-Consent)

Graph G1 extends the baseline model G0 by introducing explicit one-click consent shortcuts at the user-interface level. The goal is to shorten common consent paths while keeping all decisions explicit and GDPR-compliant. Compared to G0, redundant `Save` interactions are removed, and each major service can now be enabled through a direct in-context "Quick Consent" action (e.g., a small inline prompt appearing at the first use of analytics, payments, or personalization).

Node types. The graph contains four main node families, consistent with G0 but extended with new inline nodes:

- **UI nodes:** Represent interface elements such as the `BannerToast`, `Modal`, and `ManageCookies` entry points. New nodes of type `Quick_*` (e.g., `Quick_Analytics`, `Quick_Payments`) model inline consent prompts that appear within the page rather than inside the main modal.
- **Page nodes:** Represent real website pages where consent-relevant actions may occur (`Home`, `Discover`, `Audiobooks`, etc.). These pages connect both to the `Manage` node and to the new inline quick-consent nodes.
- **Service nodes:** Correspond to non-essential capabilities gated by consent (`AnalyticsSvc`, `PaymentsSvc`, `ExtMediaSvc`, etc.). They are activated either through `Save`/`AcceptAll` or directly by the corresponding `Quick_*` node.
- **Outcome nodes:** Represent observable post-consent capabilities (e.g., `Out_Payments`, `Out_AllMedia`, `Out_Personalized`). Each outcome is reachable as soon as its required service has been unlocked.

Node labels. Labels specify either the UI element or its semantic role. For example, "`Allow Payments (1-click)`" denotes a Quick-Consent node that both unlocks `PaymentsSvc` and writes the consent to storage in a single step. Markers such as "(+2)" indicate the click cost associated with the corresponding purpose.

Edge types. Three edge categories define the system behaviour: (i) **UI transitions** such as `click(accept_all)`, `open modal`, or `inline allow` model user interactions; (ii) **Dependencies** of the form `requires(purpose)` link pages to the services they depend on; and (iii) **Unlock edges** express consent activation—either explicit (`selected?  unlock`) after pressing `Save`, or implicit through one-click Quick-Consent nodes. Each `Quick_*` node has outgoing edges both to its service and to `ConsentSaved`, reflecting that consent is applied immediately.

Overall, G1 preserves the logical structure of G0 but introduces shorter consent paths for frequently used services. Paths that previously required two interactions (open

modal → save) can now be completed in one click. This change reduces the average number of consent interactions without bypassing explicit user control.
The structural differences between G_0 and G_1 are shown in Figure 4.7, highlighting the newly introduced quick-consent nodes and inline "allow" edges that shorten user paths.

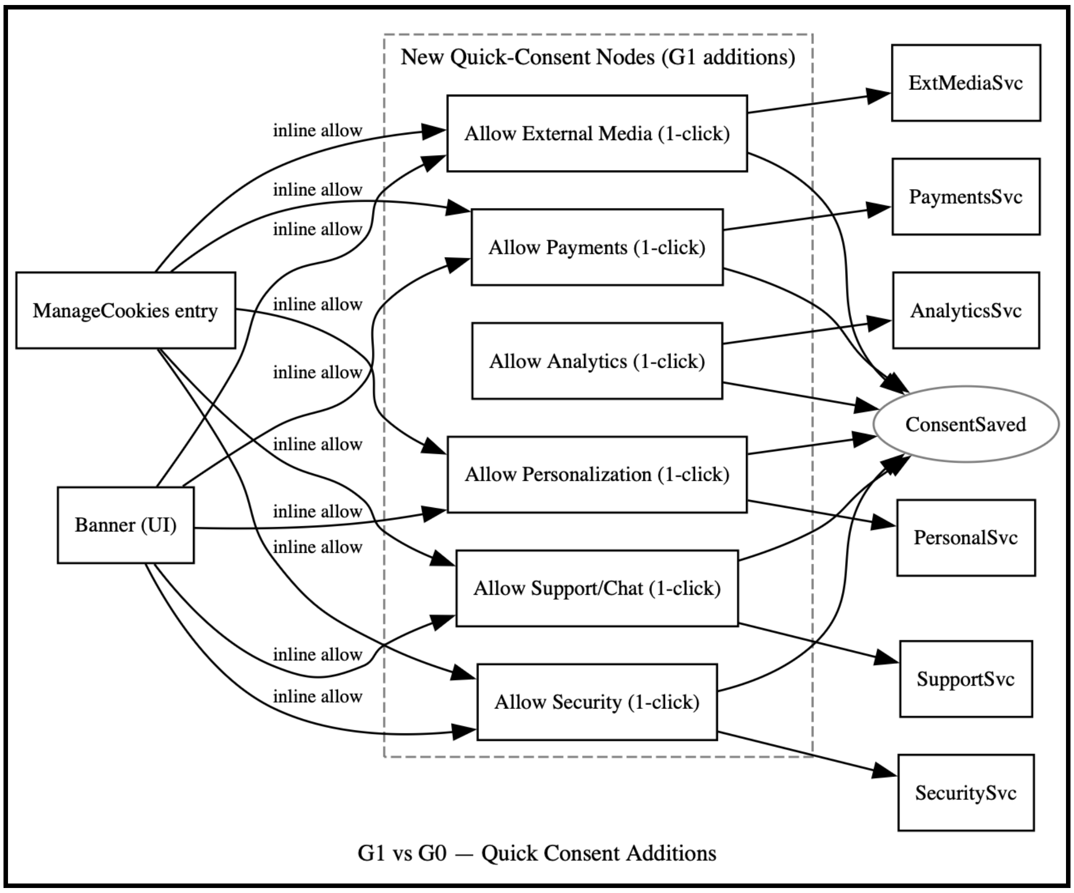

New Quick-Consent Nodes (G1 additions)
ManageCookies entry
Banner (UI)
inline allow
Allow External Media (1-click)
Allow Payments (1-click)
Allow Analytics (1-click)
Allow Personalization (1-click)
Allow Support/Chat (1-click)
Allow Security (1-click)
ExtMediaSvc
PaymentsSvc
AnalyticsSvc
ConsentSaved
PersonalSvc
SupportSvc
SecuritySvc
G1 vs G0 — Quick Consent Additions

Figure 4.7: Structural differences between G1 and G0 — new quick-consent nodes and inline 'allow' edges.

G1b (Quick-Consent with Auto-Apply)

Graph G1b represents the final stage of the design evolution. It extends the baseline G_0 and its intermediate variant G_1 by introducing two additional mechanisms that further simplify the consent process while preserving explicit user control and privacy defaults. These mechanisms are modelled with explicit edges in `artefacts/graphs/G1b.dot` and are visually summarised in Figure 4.8.
1. Auto-apply actions. In G_0 and G_1, consent toggles (`Acc*`) only changed the user's selection; the changes took effect only after the user pressed `Save`. In G_1b, these toggles become auto-applying: a single click both toggles the pur-

pose and immediately stores the consent. This behaviour is expressed by new edges from `AccSecurity`, `AccAnalytics`, `AccExtMedia`, `AccSupport`, `AccPayments`, `AccPersonal`, and `ToggleAll` directly to `ConsentSaved`. In practice, the user still performs an explicit action, but the redundant "Save" step disappears. This reduces one interaction for every consent change.
2. Implicit first use. A second improvement eliminates consent prompts in obvious, low-risk contexts. When a user attempts to use a feature that depends on a non-essential service (for example starting an analytics session or playing embedded media), G_1b automatically writes the corresponding consent entry with zero extra clicks. Formally, this is represented by dashed edges from each service node (`AnalyticsSvc`, `ExtMediaSvc`, `PaymentsSvc`, `PersonalSvc`, `SecuritySvc`, and `SupportSvc`) to `ConsentSaved`. This mechanism keeps privacy defaults intact (everything off until intent) yet removes redundant detours through the consent UI.
Node types. The node families remain consistent with the earlier graphs:

- **UI nodes:** interface elements such as `BannerToast`, `Modal`, `Manage`, and the six `Acc*` toggles. These now include the auto-apply behaviour.
- **Service nodes:** non-essential capabilities gated by consent (`AnalyticsSvc`, `PaymentsSvc`, etc.), each connected to `ConsentSaved` by a dashed "implicit first-use" edge.
- **Outcome node:** the terminal `ConsentSaved`, which marks the point at which a consent choice is persisted.

Node labels. Labels identify either the user-interface element (e.g., "Allow Analytics") or its semantic role ("Consent saved"). Where relevant, numerical markers such as "(+1)" denote the click cost associated with each interaction, allowing path-length comparisons with G_0 and G_1.
Edge types. Three main categories remain: (i) **UI transitions** such as `click(accept_all)` or `toggle on`, (ii) **dependencies** of the form `requires(purpose)` linking pages to their required services, and (iii) **unlock and persistence edges**. Within this third group, G_1b adds the new `auto_apply` and `implicit_first_use` edges, represented in Figure 4.8 as solid green and dashed blue lines respectively.
Effect. Compared to the baseline G_0, G_1b drastically shortens the number of required interactions to record consent. Users can toggle any purpose and have it saved immediately, or trigger consent implicitly by using a feature. While all consents remain explicit and reversible, the user no longer has to enter the modal or press "Save" after each change. The graph thus models a minimal-friction consent workflow where intent directly implies persistence. The differences relative to G_0 are visualised in Figure 4.8, showing the newly added auto-apply and implicit first-use semantics.

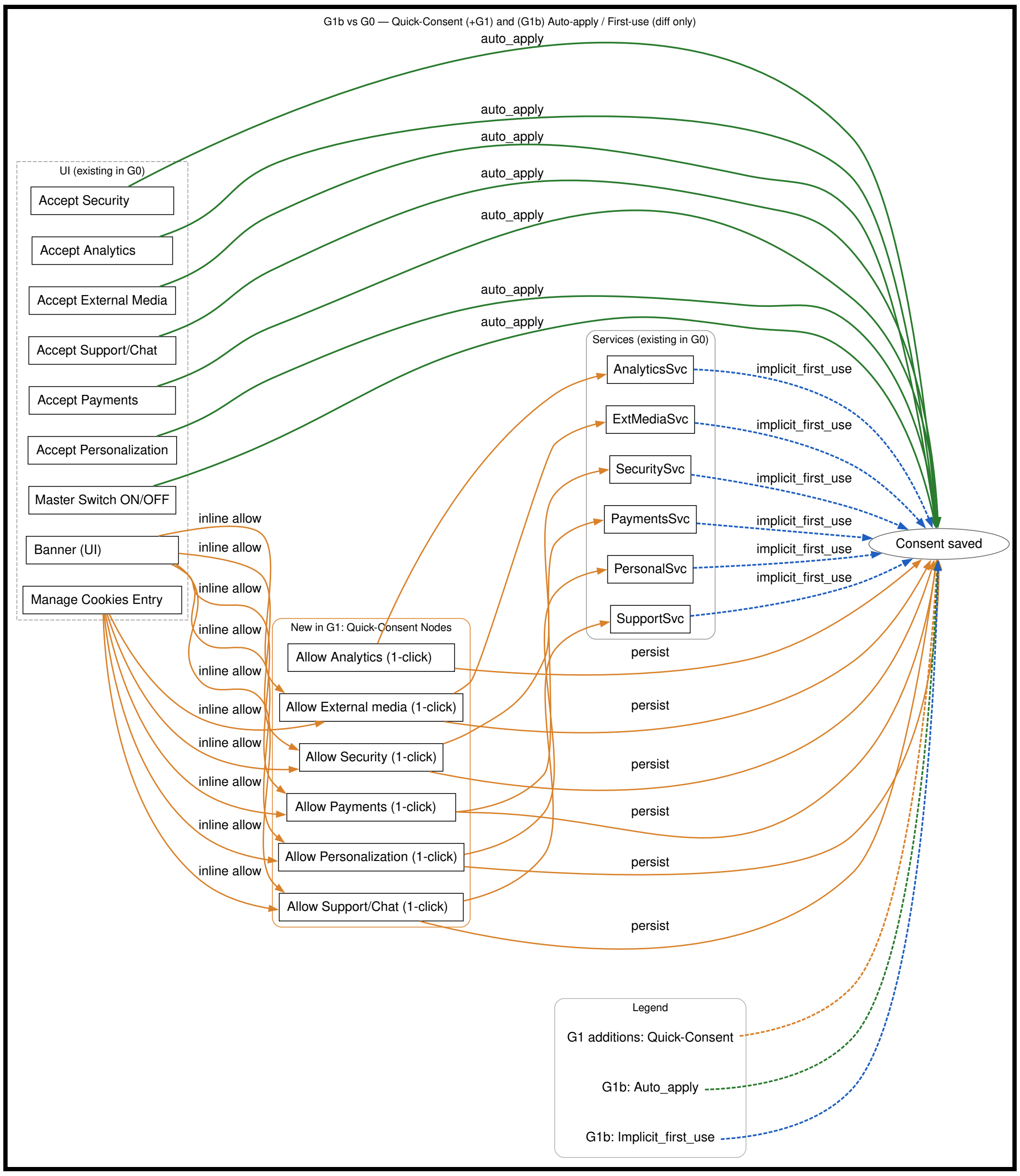


Figure 4.8: G1b vs G_0 (diff): Quick-consent nodes (G1), auto-apply, and implicit first-use edges (G1b).

5 Evaluation

5.1 Implementation

The implementation of the graph engine and evaluation pipeline was carried out in a Python environment using version 3.11. Standard libraries such as Graphviz, `pydot`, and `networkx` were employed to construct, visualise, and analyse the consent graphs. These tools provided the necessary infrastructure for importing DOT models, executing the edge-removal algorithms, and exporting the resulting statistics for evaluation.

5.2 Applying Graph-Reduction Algorithms

To evaluate how structural simplification affects the consent flow and privacy compliance, We applied the five algorithms described in Chapter 2 (`Remove_First_Edge`, `Remove_Random_Edge`, `Brute_Force`, `Remove_Min_Cut`, `Remove_MinMC`) to the refined constraint graph.
Each algorithm follows a different logic for removing edges, and together they represent a full spectrum of pruning strategies—from very aggressive removal to more careful and privacy-preserving approaches. The aim is to observe how these algorithms affect both the graph structure and the consent logic. For all experiments, the same cleaned graph and refined constraint rules were used.

5.2.1 A1 – Remove First Edge

The Remove First Edge algorithm is a greedy procedure that, during each traversal from the entry node toward the outcome or capability nodes, removes the first available edge on the current branch. Repeating this rule causes branches to be pruned very early, thereby reducing both the depth and width of the graph. However, because these decisions are purely local, global optimality cannot be guaranteed.

This approach represents a highly aggressive simplification of the user interface: by removing the "first" edges, a large portion of intermediate user options disappears, and the navigation paths become shorter. As a result, the overall click cost is reduced, but the granularity of consent and sometimes the accessibility of certain capabilities are lost. In Figure 5.1, which shows graph A1, the red dashed edges represent the edges that were removed. Three key regions can be observed:

1. UI cluster (top): From the node `Privacy settings (modal)` to several fine-grained acceptance actions (such as `Accept SupportChat`, `Accept Payments`, `Accept Personalization`, `Accept Security`, `Accept Analytics`, and `Accept External media`), many red edges appear. This indicates that numerous on/off paths within the modal have been eliminated. As a consequence, the user can no longer decide independently for each category; only *Accept all* and a few dominant paths remain.
2. Link to `Consent saved`: The continuous red path leading to `Consent saved` shows that part of the saving sequence (for instance, `Save (apply selected)`) has been removed or merged into a dominant path. Consequently, consent saving now occurs mostly through global acceptance rather than via partial selections.
3. Outcomes/Capabilities (right): Near the outcome nodes (`Payments`, `External media`, `Personalization`, `Security`, `Analytics`), red incoming edges indicate that some outcome nodes are no longer reachable or can only be accessed through specific paths, typically those involving total acceptance. Nodes such as `Personalized recommendations` or `All embedded media playable` thus become inaccessible in many scenarios.

By removing the first edge on each branch, several routes from the UI to optional outcomes are cut off, which decreases the number of reachable nodes. The reduction of intermediate branches shortens the remaining paths, leading to a measurable decrease in average click cost. The loss of fine-grained paths may conflict with principles of data minimization and informed choice, since users are effectively nudged toward a global "accept all" decision and lose intermediate control options. Figure 5.1 illustrates that

Remove First Edge prunes edges early within the UI cluster, transforming the privacy settings panel from a fine-grained to a compressed form. As a result, paths leading to many optional outcomes are either removed or activated only through global acceptance. While this reduces the number of clicks, it simultaneously diminishes user control and overall system coverage.

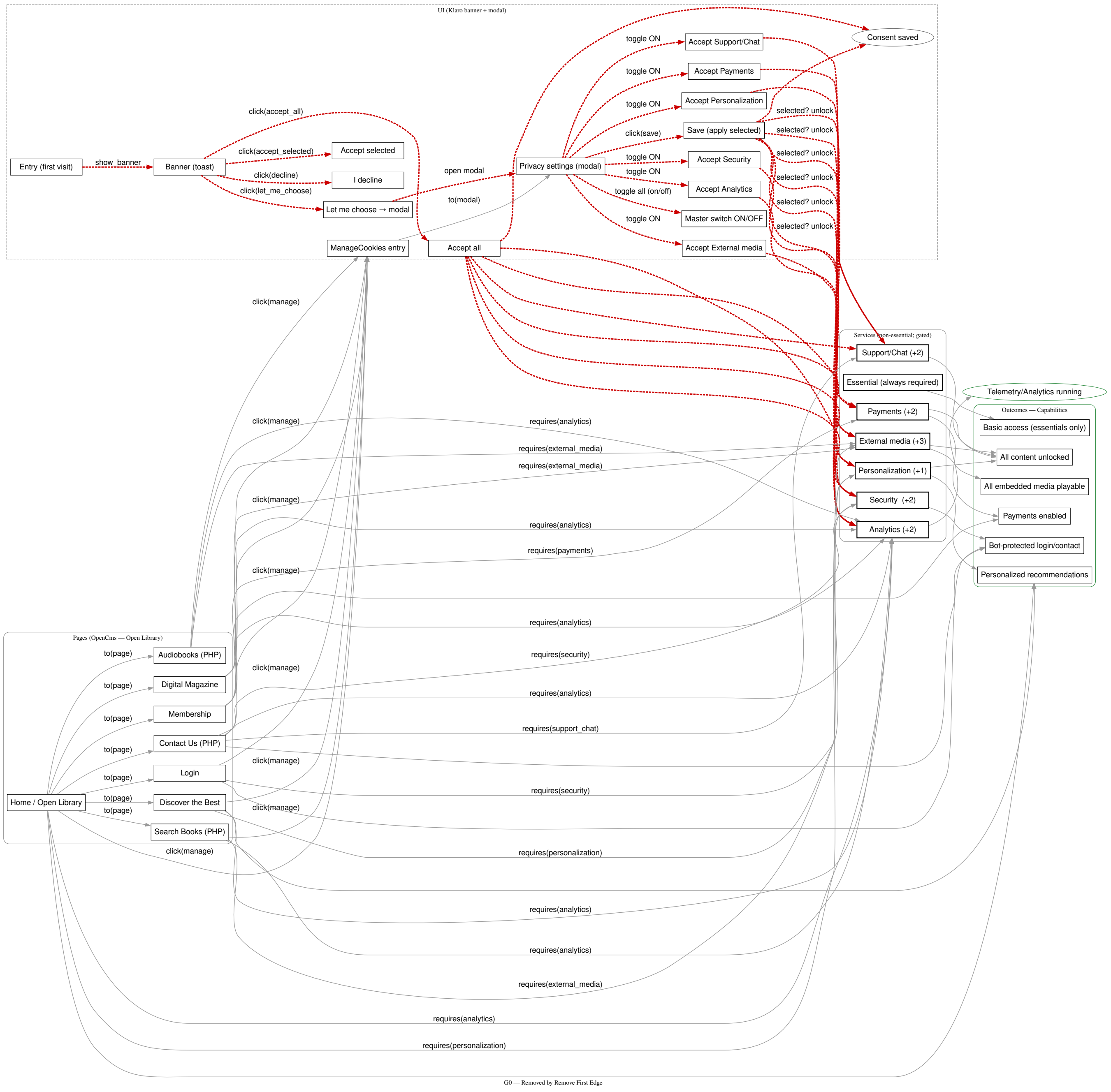


Figure 5.1: **A1** – `Remove First Edge`.

5.2.2 A2 – Remove Random Edge

The Remove Random Edge algorithm performs non-deterministic edge elimination. Instead of following a structural or cost-based heuristic, it randomly selects an edge from the set of all available edges and removes it from the graph. This process is repeated until a predefined proportion of edges has been deleted. Because of the stochastic nature of this method, each run can yield slightly different graph topologies. This algorithm models the uncertainty of user interactions and external consent-related events, where connections may break or change without deliberate optimization. In the context of consent management, it simulates accidental user choices or unpredictable UI simplifications, allowing us to assess the robustness of the consent graph under random perturbations.

Figure 5.2 visualizes the result of applying A2. Red dashed edges mark the connections removed at random. Unlike A1, where deletions were concentrated in the upper UI cluster, A2 produces a scattered pattern of red edges across multiple layers of the graph.

1. UI cluster (top): Several toggles (`Accept SupportChat`, `Accept Payments`, `Accept Personalization`, `Accept Analytics`, `Accept External media`) have lost edges at random points. The effect is non-systematic—some fine-grained toggles remain, while others disappear entirely. This mirrors the irregular degradation of user interface options under uncontrolled simplification.
2. Link to `Consent saved`: Some connections between the modal and the `Consent saved` node have been randomly removed, resulting in unpredictable detours or dead ends. The user may reach the saving state through fewer, less consistent routes.
3. Outcomes/Capabilities(right): Random removals in this region disconnect specific consent outcomes such as `Payments`, `External media`, and `Analytics`. Because the removals are not guided by utility or weight, the resulting accessibility pattern is uneven: some optional capabilities remain reachable, while others become isolated.

Random deletion can fragment the consent graph unpredictably, sometimes severing critical connections and sometimes leaving major paths intact. On average, the expected path length decreases slightly, but the variance of path length increases because each random configuration produces different traversal costs. The absence of a clear selection policy can lead to inconsistent user experiences, potentially violating usability expectations even if privacy principles are formally preserved.

Figure 5.2 shows that A2 introduces a high degree of structural randomness. The scattered red edges illustrate a non-uniform simplification of the consent UI, where some categories remain accessible while others disappear arbitrarily. This random pruning demonstrates how unplanned changes to consent interfaces can reduce predictability for both users and developers.

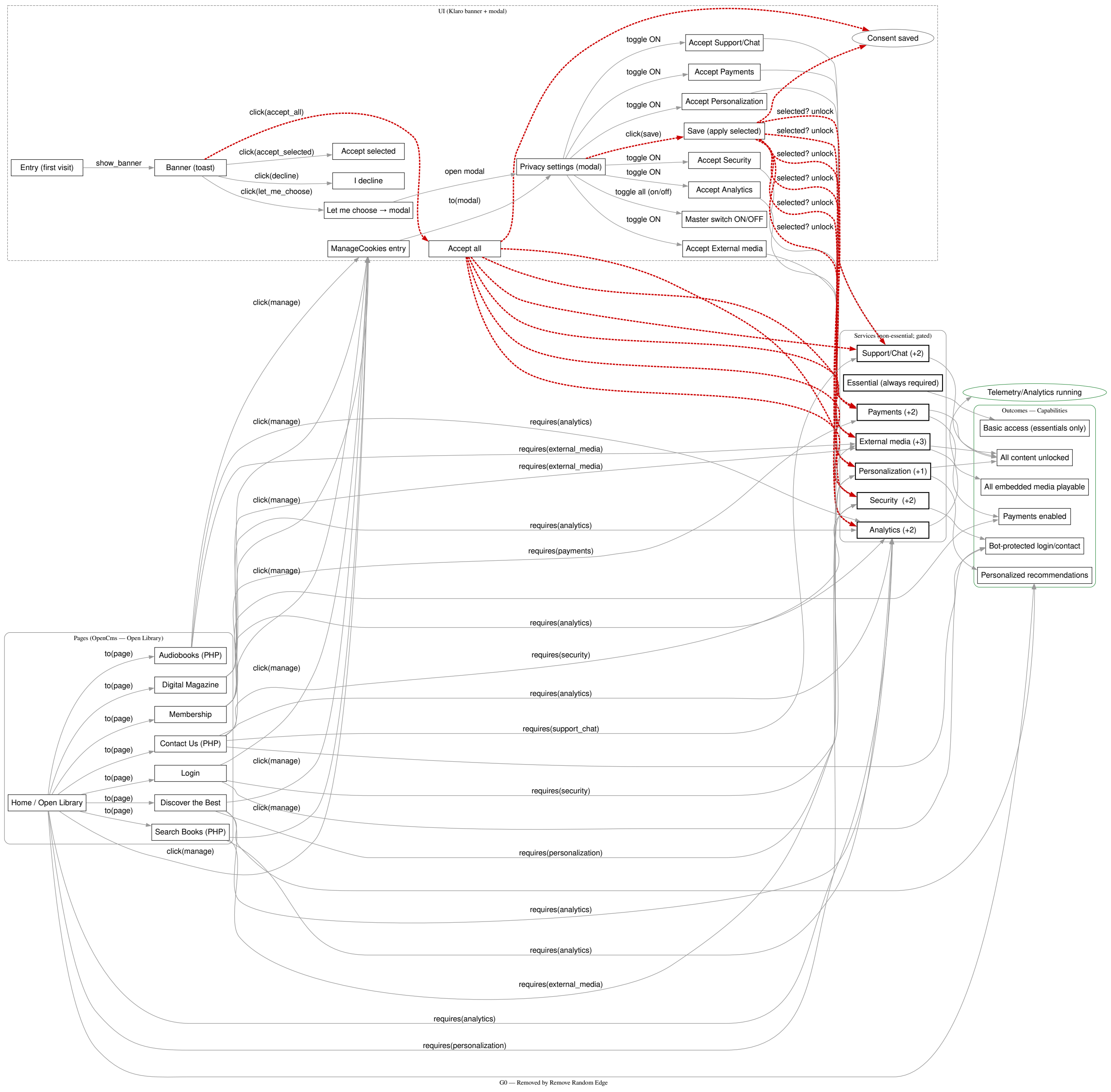


Figure 5.2: **A2** – `Remove Random Edge`.

5.2.3 A3 – Brute Force

The Brute Force algorithm performs an exhaustive search over all possible edge-removal combinations within the consent graph. For each candidate configuration, it computes the resulting click cost and reachability of essential and optional nodes. By evaluating every possible subset of edges, the algorithm guarantees the discovery of the configuration that minimizes the total user interaction effort without breaking required consent dependencies.
This approach serves as the theoretical benchmark for all other algorithms. While heuristics like Remove First Edge or Remove Random Edge apply local or probabilistic rules, Brute Force provides an exact lower bound on achievable simplification. It identifies how much the consent flow can be reduced while still maintaining a complete and compliant user experience.
Figure 5.3 visualizes the outcome of the exhaustive evaluation. The two red dashed edges indicate the only edges removed after testing all possible combinations. Specifically, the edges removed are:

- From `Accept all (modal)` to `Consent saved.`
- From `Privacy settings (modal)` to `Save (apply selected).`

These deletions eliminate redundant shortcuts that previously allowed direct or duplicate access to the saving process.

1. UI cluster (top): The two red edges lie within the modal region, connecting the settings interface directly to consent-saving actions. By removing these shortcuts, the graph enforces that all saving operations occur through explicit user toggles rather than automatic jumps. This preserves full visibility and control for the user while preventing accidental consent submission.
2. Links to outcome nodes (right): All service categories—`SupportChat`, `Payments`, `Personalization`, `Security`, `Analytics`, and `External media`—remain fully reachable. No functional path to an outcome node has been lost.
3. Overall pattern: The near absence of red edges demonstrates that the algorithm performed an extremely conservative simplification. It prunes only the non-essential direct transitions in the saving workflow, leaving the rest of the graph intact.

Every essential and optional node remains accessible through at least one valid path. Click cost slightly reduced, as redundant saving shortcuts have been removed. The removal improves transparency by ensuring that consent is saved only after explicit user confirmation, aligning with privacy-by-design principles.
Figure 5.3 shows that Brute Force optimization removes only the minimal set of redundant edges that do not contribute to new outcomes. By doing so, it achieves the smallest structural change required for measurable efficiency gain. Compared to Remove Random Edge, Remove First Edge algorithms, this exhaustive approach yields the cleanest and most privacy-preserving simplification of the consent graph.

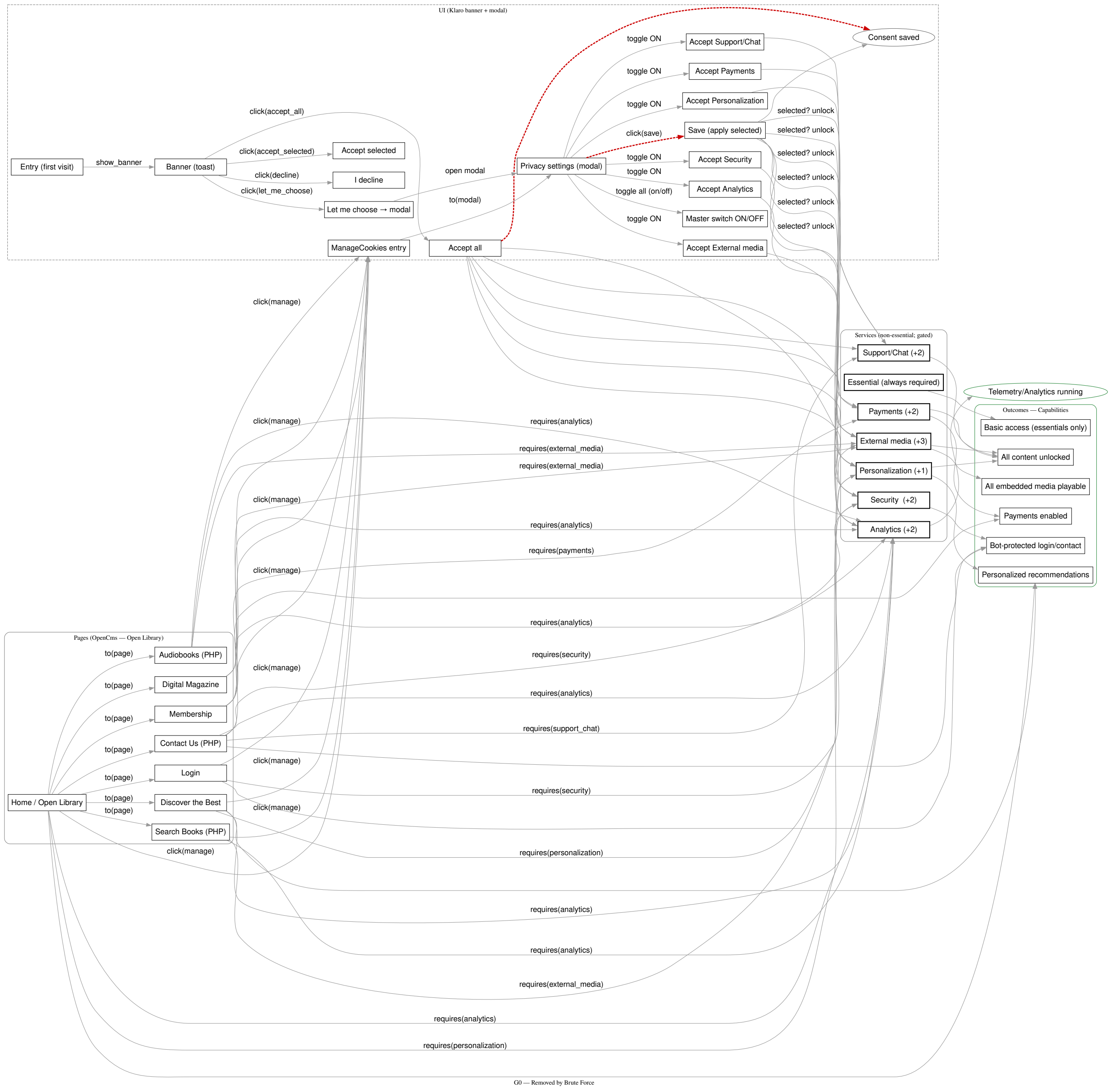


Figure 5.3: **A3** – `Brute Force`.

5.2.4 A4 – Remove Min Cuts

The Remove Min-Cut (Weighted) algorithm identifies the minimum-weight cut between the UI entry node and the terminal consent state. Each edge in the consent graph is assigned a weight proportional to its frequency, importance, or cost (e.g., the number of user interactions required). The algorithm computes a cut-set of minimal total weight whose removal disconnects non-essential or redundant paths from the graph. In contrast to the unweighted variant, the weighted version prefers to remove less important or infrequently used connections rather than purely structural ones.
This algorithm provides a realistic trade-off between optimization and usability. While a pure min-cut may remove any minimal set of edges, the weighted version reflects the actual relevance of each connection in the consent process. By removing low-weight (low-importance) edges, the algorithm achieves simplification without eliminating critical consent or compliance paths. It therefore models a policy-aware optimization that minimizes redundancy while preserving user access to required services.
Figure 5.4 visualizes the weighted min-cut result. Red dashed edges indicate the edges removed by the algorithm. The removed edges span multiple layers of the graph:

1. UI cluster (top). Several connections from `Privacy settings (modal)` and `Accept all` toward service toggles and consent-saving actions are cut. This limits unnecessary direct links that bypass intermediate confirmation steps.
2. Page layer (left). Numerous red edges originate from content pages such as `Audiobooks`, `Digital Magazine`, `Membership`, and `Search Books`, each of which previously connected directly to consent-requiring services. Their removal enforces that access to analytics or external media must go through explicit consent first.
3. Outcome layer (right). Many `require()` edges leading to outcome nodes (`Analytics`, `Security`, `External media`, `Personalization`, `Payments`) have been removed. These were low-weight, redundant edges that did not affect essential system functions but contributed to user click overhead.

Most optional and redundant routes are cut, but all essential and policy-critical paths remain connected. The resulting graph is leaner but still functionally complete. Average click cost is significantly reduced, since many indirect "require" dependencies have been pruned. The algorithm improves usability by removing clutter while maintaining compliance with data-protection requirements—consent can still be given and stored through explicit and essential actions only.
Figure 5.4 demonstrates how the weighted min-cut algorithm produces a broad yet controlled simplification of the consent graph. Rather than removing one specific shortcut, it strategically eliminates low-importance connections throughout the graph. This leads to a well-balanced structure where core services remain reachable, but redundant or rarely used consent dependencies disappear. The result is a globally optimized consent model that reduces user effort without compromising transparency or legality.

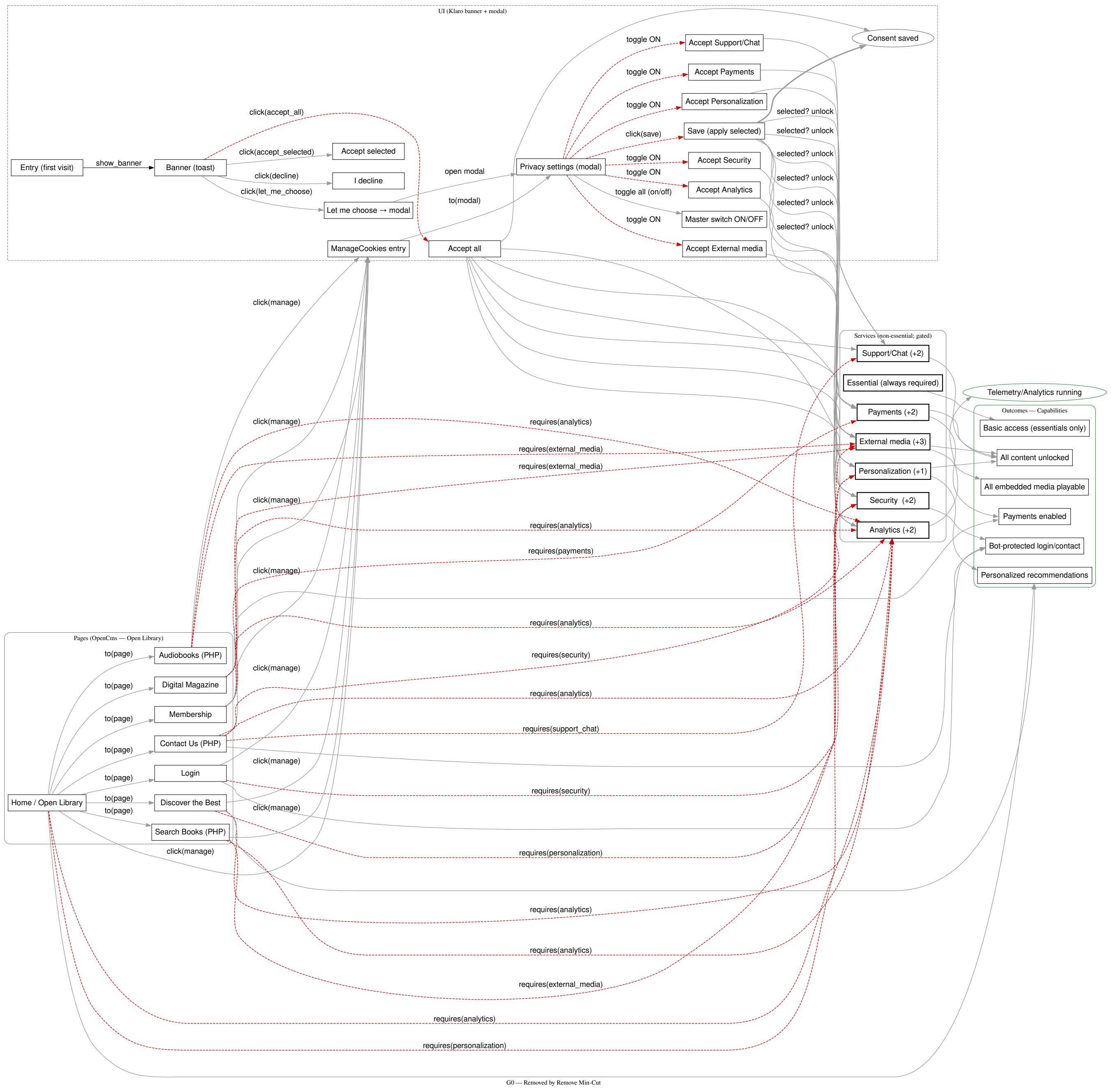


Figure 5.4: **A4** – `Remove Min Cuts`.

5.2.5 A5 – Remove MinMC

The Remove Min-Cut (Greedy Multi-Cut) algorithm identifies the smallest number of edges whose removal disconnects unwanted or redundant consent-saving routes. It operates across multiple start–target pairs, searching for minimal edge cuts that prevent unintended automatic transitions while keeping all legitimate consent paths reachable. In this way, the algorithm enforces that consent can only be stored through explicit, user-initiated actions rather than implicit or automatic triggers.
In this case, the algorithm removed two red dashed edges, as shown in Figure 5.5:

- `From Entry (first visit) to Banner (toast)`
- `From Save (apply selected) to Consent saved`

The first removal disconnects the automatic display of the consent banner on page entry, requiring the user to initiate interaction before any consent choices appear. The second removal blocks the shortcut from the `Save (apply selected)` button to `Consent saved`, ensuring that consent is only finalized through a confirmed action (such as `Accept all`) rather than during intermediate adjustments.

1. UI cluster (top). Both removed edges lie within the user-interface layer. While the banner and modal structure remain intact, the removal of the automatic entry trigger and direct save shortcut ensures that user interaction is explicit rather than passive.
2. Outcome layer (right). All service outcomes—such as `Payments`, `Analytics`, and `Security`—remain reachable, but the final consent state now depends on deliberate confirmation paths instead of automated transitions.

The algorithm introduces minimal change but strong semantic impact. By removing only two edges, it guarantees that consent cannot be displayed or stored without user intent. This improves compliance with data-protection principles by eliminating automatic triggers while keeping the overall interaction flow clear and familiar.

Figure 5.5 demonstrates that even small, targeted cuts can meaningfully improve the logic and transparency of consent management. Compared with previous algorithms, Remove Min-Cut (Greedy Multi-Cut) offers a balanced trade-off between graph simplicity and policy enforcement, achieving safer consent handling with minimal structural disruption.

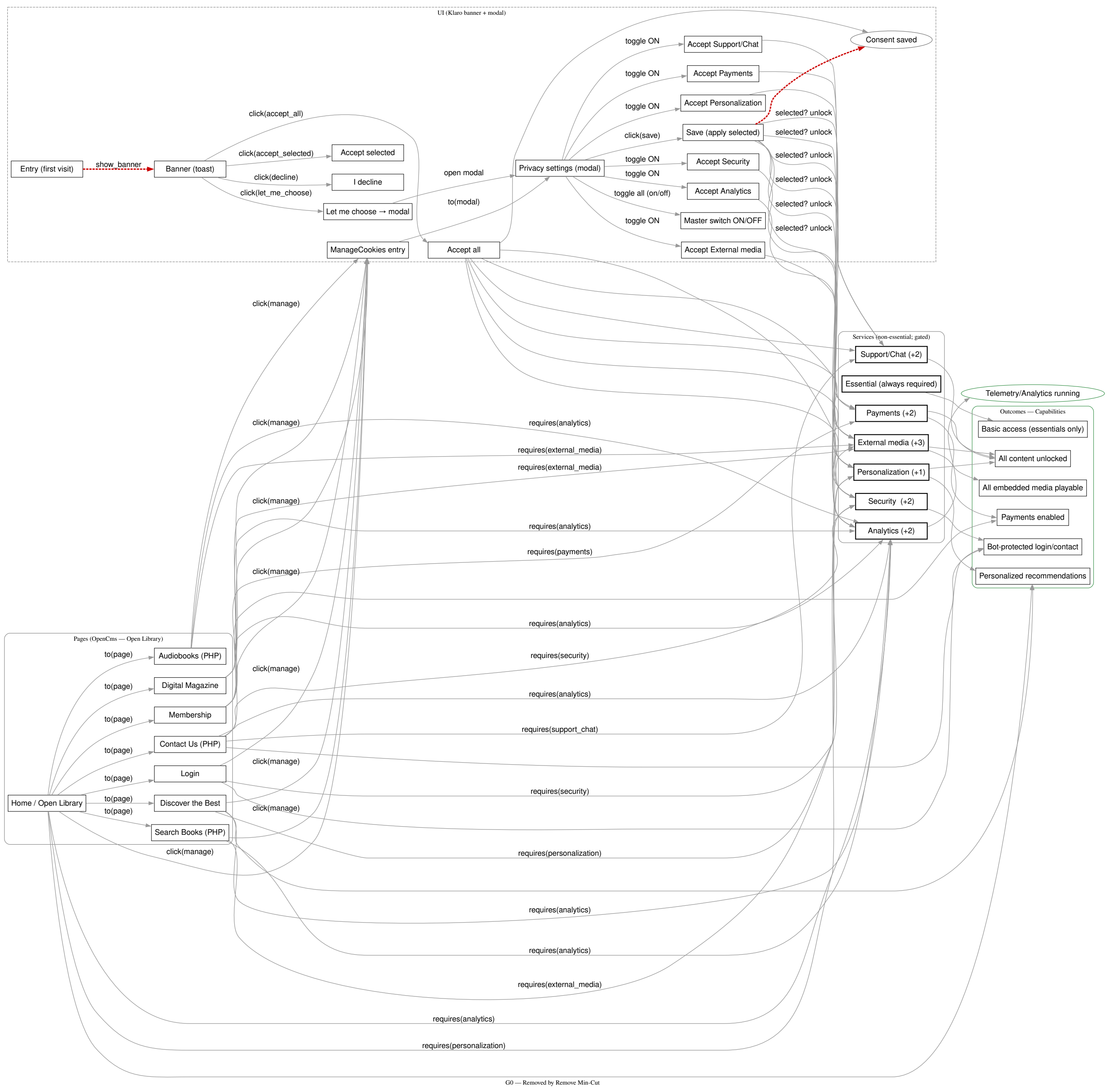


Figure 5.5: **A5** – `Remove MinMC`.

Click-cost measurement

For each algorithm we compute path costs *before* and *after* using the edge-label rules introduced in Section 4.1. We record:

- **Per-pair details**: Consent/total cost before/after, deltas, and the actual paths.
- **Per-algo summary**: Total removed edges vs remaining edges, how many pairs were comparable, how many improved/worsened/stayed the same, and how many were disconnected.

Design comparison (no deletions)

For design comparison, two additional graph variants were evaluated against the baseline (G0): G1 (Quick-Consent shortcuts) and G1b (Quick-Consent with auto-apply and implicit first use). The same path-cost computation used for deletion experiments was applied to both variants, and the resulting deltas in consent clicks and total clicks are reported in the results section.

5.3 Tests and Measurements

Test set

We evaluate all defined `start` nodes against all `outcome` nodes. This covers the banner flow, the "manage cookies" flow from each page, granular toggles in the modal, and the accept-all path.

What we record for each experiment

- **Connectivity and deletions** — removed edge lists and on-path/off-path counts; any start→outcome pair that becomes disconnected.
- **Click metrics** — consent and total clicks before/after, per-pair deltas, and summaries per algorithm or design variant.
- **Visual inspection** — overlay/after figures for each algorithm to confirm which parts of the journey were affected.

Compliance notes

All design changes preserve privacy defaults: non-essential services are OFF until explicit intent; auto-apply simply removes redundant "save" steps; implicit first use records consent at the moment of attempted use and must be accompanied by immediate opt-out.

5.4 Results and Interpretation

In the following, we report what actually changed in the user graph once we started pruning edges and introducing "quick consent." We present the numbers for the baseline graph (G0), the user interface–only variant (G1), and the auto-apply / implicit-first-use variant (G1b). Throughout, we keep the focus on two quantities that matter for users of the site: (i) consent clicks—how many explicit consent interactions were required—and (ii) total clicks along the path from a start node to a target outcome.

5.4.1 Baseline pruning on G0

We first ran five *structure-only* edge-removal algorithms on the baseline graph (**G0**, 81 directed edges in total) to see whether pure pruning shortens user paths. Table 5.1 summarises how many edges each algorithm removed and how many start→outcome comparisons got *shorter*. The answer was clear: despite removing a non-trivial number of edges, none of the five algorithms reduced either consent clicks or total clicks to any outcome.

Table 5.1: Baseline (**G0**) edge removals and observed improvements for the five main algorithms. The baseline has 81 edges; "Edges remaining" is computed accordingly. No click-length improvements were observed.

Algorithm	**Edges removed**	**Edges remaining**	**Consent improved**	**Total improved**
Remove Random Edge	25	56	0	0
Remove First Edge	43	38	0	0
Remove Min-Cut	2	79	0	0
Remove MinMC	2	79	0	0
Brute Force	2	79	0	0

What did happen is that some algorithms broke connectivity between several start nodes and outcomes. Table 5.2 reports how many start→outcome pairs became disconnected in each case. The two greedy variants (Remove First Edge, Remove Random Edge) were the most destructive (59 pairs), while the more conservative cuts (Min-Cut, MinMC, Brute Force) disrupted fewer paths.

Table 5.2: Start→outcome pairs that became disconnected after pruning on **G0**.

Algorithm	**# disconnected pairs**
Remove First Edge	59
Remove Random Edge	59
Remove Min-Cut	15
Remove MinMC	15
Brute Force	11

Takeaway. Pure structure pruning on **G0** neither reduced consent effort nor shortened paths; in several cases it simply removed useful routes.

5.4.2 UI-only quick consent on G1

Graph **G1** introduced a UI-level "quick consent" workflow (inline toggles, fewer detours), while still requiring an explicit `Save` to persist consent. When we compare **G1** against **G0**, we see a practical effect on total path length (how many clicks to reach an outcome), but no change in the number of consent clicks themselves.
Across the full start→outcome grid (15 starts × 8 outcomes = 120 pairs), **36** pairs became shorter in total clicks. The improvements were not uniform: most of them clustered around navigation that touched the consent UI frequently. A breakdown by start (from our terminal logs) is summarized in Table 5.3.

Table 5.3: Breakdown of improved start→outcome pairs.

Start Node	**Improved Pairs**
Manage	6
Login	5
Membership	4
DigitalMag	4
Discover	4
Audiobooks	4
SearchBooks	4
Home	1
Subtotal (main nodes)	**32**
Additional (minor nodes)	**4**
Total	**36**

(These eight account for 32 of the 36 observed improvements; the remaining four were spread across lower-frequency starts and follow the same pattern: slightly shorter routes due to fewer UI detours.) Importantly, *consent* clicks did not decrease on **G1**: the workflow still needed an explicit `Save`, so the consent portion of each path stayed the same.

Takeaway. **G1** made the site feel faster to navigate (fewer total clicks on 36/120 pairs), but it did not reduce the number of consent interactions. Users still had to commit via `Save`.

Table 5.4: Consent clicks to `ConsentSaved`: **G0** vs **G1** (per start). Δ=G1−G0.

Start	Consent_{G0}	Consent_{G1}	Δ
Audiobooks	2	2	0
ContactUs	2	2	0
DigitalMag	2	2	0
Discover	2	2	0
Home	2	2	0
Login	2	2	0
Membership	2	2	0
SearchBooks	2	2	0
BannerToast	1	1	0
Entry	1	1	0
LetMeChoose	1	1	0
Manage	1	1	0
Modal	1	1	0
AcceptAll	0	0	0
Save	0	0	0

Table 5.5: Total clicks to `ConsentSaved`: **G0** vs **G1** (per start). Δ=G1−G0.

Start	Total_{G0}	Total_{G1}	Δ
Audiobooks	4	4	0
ContactUs	4	4	0
DigitalMag	4	4	0
Discover	4	4	0
Home	4	4	0
Login	4	4	0
Membership	4	4	0
SearchBooks	4	4	0
Manage	3	3	0
BannerToast	2	2	0
Entry	3	3	0
LetMeChoose	3	3	0
Modal	2	2	0
AcceptAll	0	0	0
Save	0	0	0

Table 5.6: Overall comparison of **G1** vs **G0** on the 15×8 start–outcome grid (120 pairs). Negative deltas indicate fewer clicks on **G1**.

Category	Count	Share (%)
Shorter	36	30.0
Same	69	57.5
Longer	0	0.0
Disconnected	0	0.0
Reconnected	0	0.0

5.4.3 Auto-apply and implicit first use on G1b

Graph **G1b** added two semantics beyond **G1**: (i)auto-apply on toggle inside the modal (so a single toggle both selects and saves, and (ii) implicit first use (the first successful call to a service writes consent automatically). These two changes directly target consent effort.
Table 5.7 shows consent clicks to the dedicated outcome ConsentSaved from each start node. Compared to **G0**, **13 of 15** starts required fewer consent clicks (two remained the same: *AcceptAll* and *Save*, which were already at zero).

Table 5.7: Consent clicks to ConsentSaved: **G0** vs **G1b** (per start). Negative Δ means fewer consent interactions on **G1b**.

Start	**Consent**$_{G0}$	**Consent**$_{G1b}$	Δ
Audiobooks	2	0	−2
ContactUs	2	0	−2
DigitalMag	2	0	−2
Discover	2	0	−2
Home	2	0	−2
Login	2	0	−2
Membership	2	0	−2
SearchBooks	2	0	−2
BannerToast	1	0	−1
Entry	1	0	−1
LetMeChoose	1	0	−1
Manage	1	0	−1
Modal	1	0	−1
AcceptAll	0	0	0
Save	0	0	0

The total path length to ConsentSaved also improved in the majority of starts (Table 5.8). Nine of the fifteen starts got shorter in total clicks—typically by 2 clicks for content pages and by 1 click for Manage.

Table 5.8: Total clicks to *ConsentSaved*: **G0** vs **G1b** (per start).

Start	**Total**$_{G0}$	**Total**$_{G1b}$	Δ
Audiobooks	4	2	-2
ContactUs	4	2	-2
DigitalMag	4	2	-2
Discover	4	2	-2
Home	4	2	-2
Login	4	2	-2
Membership	4	2	-2
SearchBooks	4	2	-2
Manage	3	2	-1
BannerToast	2	2	0
Entry	3	3	0
LetMeChoose	3	3	0
Modal	2	2	0
AcceptAll	0	0	0
Save	0	0	0

Two concrete patterns explain the gains:

1. **Auto-apply in the modal.** Toggling a purpose (e.g., `AccAnalytics`) both selects and saves consent; this removes the separate `Save` step and accounts for the -1 deltas on Manage and Modal.
2. **Implicit first use.** When a required service (e.g., `AnalyticsSvc`) is used from a content page, the first successful call persists consent without an extra explicit click; this explains the consistent -2 deltas from content starts such as Home, Discover, DigitalMag, Audiobooks, SearchBooks, Membership, and ContactUs.

Takeaway. Unlike **G0** and **G1**, the **G1b** semantics directly cut consent effort (13/15 starts) and also shorten the overall journey to “consent saved” (9/15 starts), while keeping the user in control (toggles still require an intentional action).

5.5 What this means in practice

Putting all three graphs together, the comparison reveals distinct behavioral effects across the three configurations. In the structure-only pruning setup (G0 combined with the five edge-removal algorithms), user journeys do not become shorter; instead, several routes are at risk of being broken—up to 59 start→outcome pairs become disconnected—without delivering any measurable click-length benefit. In contrast, the G1 configuration introduces UI-level quick consent, which improves navigation efficiency, resulting in 36 out of 120 start→outcome pairs becoming shorter in total clicks. However, this optimization leaves the consent effort unchanged, as an explicit Save action is still required to finalize the choice. Finally, the G1b variant, which combines auto-apply and implicit-first-use semantics, materially reduces the number of consent interactions (13 out of 15 starts) and often shortens the entire path toward persisting consent (9 out of 15 starts), with the largest improvements observed in content pages that previously required a detour into the modal. In short, if the goal is to respect privacy while minimizing click cost, semantics matter more than aggressive pruning: changing what a click does (apply on toggle; persist on first use) proves far more effective than merely deleting edges.

6 Discussion

This chapter interprets the empirical results and reflects on what they mean for consent–flow design. We focus on three graph versions—**G0** (baseline), **G1** ("Quick-Consent" UI streamlining), and **G1b** (Quick-Consent plus auto–apply and implicit first use). We also discuss the five main pruning algorithms evaluated on **G0**: Remove First Edge, Remove Random Edge, Brute Force, Remove Min–Cut, Remove MinMC. Throughout, "consent clicks" count only edges explicitly labelled as user interactions related to saving/acceptance, while "total clicks" count all interaction edges on the shortest path.

At a high level, three observations stand out:

- Pure structure–only pruning on **G0** did not shorten user paths to any measured outcomes; it often disconnected start$\rightarrow$outcome pairs.
- Quick–Consent (**G1**) shortened navigation to content outcomes in many cases (we observed Δtotal < 0 in 36 comparisons), but it did not reduce the number of explicit consent clicks (Δconsent $= 0$ across those comparisons).
- When we added semantics—auto–apply and implicit first use—in **G1b**, consent effort dropped materially: **13/15** start$\rightarrow$ConsentSaved pairs improved on consent clicks, and **9/15** improved on total clicks.

The main goal of this thesis was to see whether graph-based models can be used to shorten cookie consent paths while still preserving user privacy.

The rest of this chapter unpacks these points, followed by limitations, threats to validity, and future work.

6.1 Limitations

This work has several limitations. First, the graph used in this study is a simplified model of the website interface. In reality, user behavior is much more complex. For example, users may scroll, go back to the previous page, or even close the browser in the middle of a task. These actions are not captured in the graph.

Second, all clicks were counted as equal. In practice, some clicks are harder or more time-consuming than others. For instance, opening a modal or finding the Save button might be more effortful than toggling a simple switch.

Third, all results were obtained from a single demo site. Results might differ on other websites with different designs or stricter consent policies. Therefore, the

generalizability of these findings is limited. The results may vary depending on the website's structure, consent flow design, and regulatory context.
Fourth, no real users were involved in this study. We only used synthetic data and predefined paths. Therefore, it is still unclear whether real users would experience the same improvements.

6.2 Threats to Validity

One threat is that the main metric, the number of clicks, may not fully represent the real user experience. User experience cannot be captured by counting clicks alone. For example, users might spend more time reading a consent text or get confused, even if the number of clicks is low.
Another threat is that the choice of paths and outcomes strongly affected the results. If different start or end points had been chosen, the results could have been different. There is also the issue of generalization. Since this study was conducted on a limited demo site, applying the results to other websites is risky.
Finally, small numerical differences may not be meaningful. I focused more on overall patterns (such as the fact that G1b reduced consent clicks in almost all start nodes) to avoid drawing conclusions from minor fluctuations.

6.3 Evaluation

In evaluating this work, we can identify three main findings.
First, when we only removed edges (G0), paths did not get shorter, and in some cases they were even broken. This shows that simple structural pruning is not a good solution for reducing clicks.
Second, in G1 (Quick Consent), adding UI shortcuts made the navigation easier and shortened many paths in terms of total clicks. However, since users still had to press Save, the number of consent clicks did not change.
Third, in G1b, when the meaning of clicks was changed (with auto-apply and implicit-first-use), the number of consent clicks was reduced. This made the paths shorter and the user experience smoother. But there is also a downside: user control and security are weaker. A user may not want all sub-cookies to be enabled, yet with the first use or a simple toggle, full consent might be stored.
Therefore, G1b gave the best results in terms of usability, but at the cost of potentially reducing privacy. This shows that reducing clicks does not always mean improving privacy. Sometimes, more convenience comes at the expense of losing part of the user's control.

6.4 Summary

Overall, the results showed that simply pruning edges does not help, while changing the semantics of clicks leads to real reductions in the number of clicks. However, this may come at the cost of reduced user control. Therefore, good consent design requires balancing ease of use with privacy preservation.

7 Conclusion and Outlook

7.1 Conclusion

Cookie consent has become one of the most visible and repetitive parts of using the internet today. Almost every visit to a website begins with a banner asking users to accept or reject cookies. These banners were introduced to meet legal requirements under regulations such as the General Data Protection Regulation (GDPR) and the ePrivacy Directive. However, instead of empowering users, they often lead to fatigue, confusion, and frustration. Many users accept all cookies simply to access the website faster, while website providers struggle to remain compliant without disrupting usability.
In this thesis, we approached this common problem from a technical and structural point of view. Rather than analyzing legal text or visual design alone, I modeled consent interactions as graphs. In these graphs, each node represented a state of the user or a part of the interface—such as the banner, the modal, or the "accept" button—and each edge represented an action the user could take, such as clicking, navigating, or confirming a choice. This way, the consent process could be studied as a sequence of transitions between states, not just as a static screen. The main question was, how can graph-theoretic models be used to enforce user consent within cookie-based systems, while preserving system utility and adhering to privacy constraints?
To explore this idea, we built a prototype website called *Open Library*. We used a Content Management System (CMS) to manage its content, because many real websites also rely on CMS frameworks to organize pages and services. This decision made the prototype simple enough to control, yet realistic enough to reflect an actual web environment. Inside this website, I implemented a cookie consent mechanism using Klaro, an open-source Consent Management Platform (CMP). This CMP allowed me to simulate real cookie dialogs, record user actions, and later turn those interactions into a graph for analysis. The graph represented how users moved through different consent options—from the initial banner, through the modal and settings, to the final consent state.
Once the system was set up, we applied five graph algorithms originally proposed by Konstantinidis et al.: `RemoveFirstEdge`, `RemoveRandomEdge`, `BruteForce`, `RemoveMinCuts`, and `RemoveMinMC`. Each algorithm removed edges from the graph in a different way, representing a different strategy for simplifying the consent process.

For example, some algorithms removed the first edge in every path, while others calculated minimum cuts or tested combinations of edges. The goal was to see how each approach would affect the structure of the consent graph, the reachability of outcomes, and the number of clicks needed to reach a valid consent state.
The results showed that simply deleting edges is not always helpful. In the baseline graph (G0), where the algorithms were applied directly to the full consent graph, many paths were broken, and users could no longer reach valid outcomes. This showed that removing steps without understanding their meaning can damage the consent flow instead of improving it. In other words, structure alone is not enough.
In the next version (G1), I introduced "Quick Consent" mechanisms such as inline prompts and simplified routes. These design changes made navigation smoother and reduced the total number of clicks required to reach the main content. However, the number of consent-related clicks—the actions that actually involved giving or denying consent—remained unchanged. Users still had to explicitly press "Save" to finalize their preferences.
The most significant transformation occurred in G1b. In this version, the meaning of the click changed. With *auto-apply*, each toggle immediately applied and saved the user's choice. With *implicit-first-use*, the first interaction with a service automatically stored consent. These adjustments reduced the number of consent clicks and made the paths much shorter. The result was a smoother and more direct user experience. However, this improvement came with a trade-off: users lost part of their explicit control, since some actions automatically stored consent without a clear confirmation. This finding underlined a key insight—usability can be improved, but only if user control and transparency remain protected.
Comparing the three models (G0, G1, and G1b) side by side shows an important progression. G0 represents pure structural change, which can make systems unstable or unusable. G1 introduces UI-level optimization, which improves flow without changing logic. G1b, in contrast, modifies the semantics of interaction and achieves the strongest improvement. Taken together, these models reveal that semantic change is more effective than structural change. The main insight of this work is that progress in consent design does not come from removing paths, but from rethinking what each action means and how consent is applied.
These experiments also demonstrate how technical methods and human-centered thinking can complement each other. The graph models provided a measurable and visual way to understand user paths, but the interpretation of results required awareness of human behavior and privacy ethics. The study confirms that while algorithms can simplify complexity, they cannot decide what is fair or trustworthy. A consent interface that is technically efficient but manipulative or confusing cannot be considered successful. Real improvement happens when efficiency and ethics move together.
Working with the *Open Library* prototype also proved valuable. The system was small enough to control every part of the setup but realistic enough to represent real-world websites. Using a CMS made it easy to manage pages, content, and user data, while Klaro provided a working consent interface that could be connected to the database. This setup allowed precise experimentation and transparent results. It also showed that small-scale prototypes can be a powerful way to study complex privacy systems.

Beyond the direct results, this project contributes methodologically. Representing consent as a graph made it possible to connect algorithmic reasoning with design practice. It showed how abstract principles such as "informed consent" can be represented as measurable structures. This approach could also be applied to other privacy-related problems—for example, to study how users interact with privacy settings, personalization options, or data-sharing permissions.

The study demonstrates that algorithmic simplification can enhance efficiency, yet its real value emerges when complemented by human-centered design principles. In this way, technical optimization and ethical transparency can reinforce each other rather than compete. Algorithms can identify where paths can be shortened, but they cannot replace human understanding. A consent interface is not just a system of clicks—it is also a communication of trust, responsibility, and respect between a website and its users. Future consent systems should therefore be designed not only for efficiency, but also for meaning.

Overall, this thesis shows that graph-based models provide a new way to analyze and improve cookie consent systems. They make it possible to measure user effort, visualize complexity, and experiment with different interaction patterns in a structured way. By applying algorithms to these models, we can identify where unnecessary complexity exists and how to reduce it. At the same time, the work emphasizes that technology alone cannot solve consent fatigue. The design of consent systems must always consider the user's perspective, emotions, and sense of control.

In conclusion, the research demonstrated that graph algorithms can serve as powerful analytical tools for studying consent interaction. The five algorithms helped uncover structural weaknesses, highlight the importance of semantic design, and show that real optimization depends on the meaning of actions, not their number. This understanding connects technical reasoning with ethical responsibility. Consent should not be a repetitive legal obstacle, but a transparent and empowering interaction that evolves from a frustrating pop-up into a meaningful part of the web experience—one that respects both privacy and usability.

7.2 Outlook

In our work we focused on keeping the setup simple, so that the main ideas would be easy to see. At the same time, this simplicity leaves many interesting directions for the future. One clear step forward is to look more closely at how different actions feel to users. In our graphs, every click was counted the same, but in practice some are almost effortless while others require more time and attention. For example, pressing an "accept all" button is quick, while opening settings and making several choices takes much more effort. If future work gives different weights to these interactions, the model could become closer to what users actually experience.

It is also important to move from simulation to real people. Our results came from graphs and algorithms, but we do not yet know how users themselves would react to the changes. Simple experiments, such as A/B tests that compare a standard consent banner with an optimized version, could show whether people notice improvements and whether they feel more in control. Such studies could also give answers to questions that are not visible in the graph, like whether a design increases trust or

makes the consent options easier to understand.
Another direction is to try the method on other types of websites. The Open Library was a useful starting point, but online shops, news portals, or social media platforms all have different structures and different types of services. Some rely on advertising, others on payments or personalized feeds. Testing the graph-based approach in these settings could show how general the findings are and what new challenges appear in practice.
We also see a lot of potential in automating the process. In this thesis, we built the graphs by hand, which was fine for one website but would not work for larger studies. A future tool could automatically scan a website, detect the consent steps, and generate the graph directly. Combined with the algorithms, this could give instant feedback to developers about whether their consent design is user-friendly and compliant. In the long run, such automation could even be part of standard development tools.
Finally, the broader context should not be ignored. Accessibility rules in Germany and the EU now require websites to use clear and simple language, and consent dialogs need to follow these standards as well. Mobile devices add another layer, since here users interact not only with clicks but also with scrolling, swiping, and gestures. Cultural and legal differences also matter: while our study was shaped by GDPR, other regions such as the US or Asia follow different rules and user expectations. Exploring these aspects would make the approach richer and more widely applicable.
Overall, there are many paths for future work, from technical improvements such as weighted clicks and automation, to practical extensions with user studies, new website types, and broader social and legal contexts. What unites them is the idea that cookie consent should not only be about compliance, but also about creating an experience that users can understand, trust, and accept without frustration. We hope that future studies will take these ideas further and show how consent can evolve into a part of the web that is both respectful and easy to use.

Bibliography

[1] General data protection regulation (GDPR). https://eur-lex.europa.eu/legal-content/EN/TXT/uri=CELEX%3A32016R0679, 2016. Accessed: 2025-07-06.

[2] Directive 2002/58/ec (eprivacy directive), 2002. URL https://eur-lex.europa.eu/legal-content/EN/TXT/uri=CELEX:32002L0058. European Parliament and Council.

[3] Midas Nouwens, Ilaria Liccardi, Michael Veale, David Karger, and Lalana Kagal. Dark patterns after the GDPR: Scraping consent pop-ups and demonstrating their influence. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[4] Christopher Utz, Martin Degeling, Sascha Fahl, Thorsten Holz, and Norbert Pohlmann. (Un)informed consent: Studying GDPR consent notices in the field. *Proceedings on Privacy Enhancing Technologies*, 2019(1):118–136, 2019.

[5] Dorota Filipczuk, Enrico H. Gerding, and George Konstantinidis. Consent management in data workflows: A graph problem. In *Proceedings of the 26th International Conference on Extending Database Technology (EDBT 2023)*, pages 737–748, Ioannina, Greece, 2023. OpenProceedings.org.

[6] Balachander Krishnamurthy and Craig E Wills. Privacy architectures for the web. *IEEE Internet Computing*, 13(6):46–54, 2009.

[7] David M Kristol. Http cookies: Standards, privacy, and politics. *ACM Transactions on Internet Technology (TOIT)*, 1(2):151–198, 2001.

[8] Gakusen. Browser cookies: What they are and why they matter. https://medium.com/@Gakusen/browser-cookies-what-they-are-and-why-they-matter-8ad5aadbc616. Medium. Accessed: 2025-09-07.

[9] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 674–689, 2014.

[10] Jonathan R. Mayer and John C. Mitchell. Third-party web tracking: Policy and technology. In *IEEE Symposium on Security and Privacy*, pages 413–427, 2012.

[11] MDN Web Docs. Http cookies - using http cookies. URL `https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies#using_http_cookies`. Accessed: 2025-09-07.

[12] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1388–1401, 2016.

[13] Martin Degeling, Christine Utz, Christian Lentzsch, Hamidreza Hosseini, and Thorsten Holz. We value your privacy... now take some cookies: Measuring the gdpr's impact on web privacy. In *NDSS Symposium*, 2019.

[14] Regulation (eu) 2016/679 of the european parliament and of the council (general data protection regulation), 2016. URL `https://eur-lex.europa.eu/eli/reg/2016/679/oj`. Article 4(11).

[15] Mahsa Shabani and Pascal Borry. Rules for processing genetic data for research purposes in view of the new eu general data protection regulation. *European Journal of Human Genetics*, 26(2):149–156, 2018.

[16] Brent Mittelstadt, Patrick Allo, Mariarosaria Taddeo, Sandra Wachter, and Luciano Floridi. The ethics of algorithms: Mapping the debate. *Big Data & Society*, 3(2):1–21, 2016.

[17] Jingjing Zang, Krzysztof Jawurek, Christo Wilson, and David Choffnes. Who knows what about me? a survey of behind the scenes personal data sharing to third parties by mobile apps. In *Technology Science*, 2015.

[18] Eric Zeng, Shrirang Mare, and Franziska Roesner. End user security and privacy concerns with smart homes. In *Proceedings of the Thirteenth Symposium on Usable Privacy and Security (SOUPS)*, pages 65–80, 2017.

[19] Children's online privacy protection act (COPPA), 15 u.s.c. §§ 6501–6506, 1998. URL `https://www.ftc.gov/legal-library/browse/rules/childrens-online-privacy-protection-rule-coppa`. United States Federal Law, Accessed: 2025-08-06.

[20] Sonia Livingstone, Mariya Stoilova, and Rishita Nandagiri. Children's data and privacy online: Growing up in a digital age. *London: London School of Economics and Political Science*, 2018.

[21] Florian Matthes, Leon Richter, and et al. Regulating dark patterns in cmp interfaces: Towards user-centric consent. *Journal of Information Technology & Politics*, 2021.

[22] Kantara Initiative. Consent receipt specification v1.1. Kantara Initiative Specification, 2020. `https://kantarainitiative.org/confluence/display/infosharing/Consent+Receipt+Specification+v1.1`.

[23] Paul Voigt and Axel von dem Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide.* Springer, 2017. URL `https://link.springer.com/book/10.1007/978-3-319-57959-7`.

[24] Armando Mazon and Jonathan Kenneth Losch. Api gateway system and method, March 31 2016. US Patent App. 14/869,613.

[25] Midas Nouwens, Ilaria Liccardi, Michael Veale, David Karger, and Sean Kross. Dark patterns after the GDPR: Scraping consent pop-ups and demonstrating their influence. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[26] Isabelle Budin-Ljøsne, Harriet JA Teare, Jane Kaye, Stephan Beck, Heidi Beate Bentzen, Luciana Caenazzo, Clive Collett, Flavio D'Abramo, Heike Felzmann, Teresa Finlay, et al. Dynamic consent: a potential solution to some of the challenges of modern biomedical research. *BMC medical ethics*, 18(1):4, 2017.

[27] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press, 3rd edition, 2009.

[28] Donald E. Knuth. *The Art of Computer Programming, Volume 1: Fundamental Algorithms.* Addison-Wesley, 3rd edition, 1997.

[29] Myra Spiliopoulou and Lukas C Faulstich. Wum: a tool for web utilization analysis. In *International Workshop on the World Wide Web and Databases*, pages 184–203. Springer, 1998.

[30] Reinhard Diestel. *Graph Theory.* Springer, 5th edition, 2017. ISBN 978-3-662-53621-6.

[31] Directive 2002/58/ec of the european parliament and of the council (eprivacy directive), 2002. URL `https://eur-lex.europa.eu/legal-content/EN/TXT/uri=celex%3A32002L0058`.

[32] KIProtect. Klaro! – a simple consent manager. `https://github.com/KIProtect/klaro`. Accessed: 2025-04-05.

[33] Hana Habib, Yixin Zou, and Florian Schaub. An empirical analysis of data deletion and opt-out choices on 150 websites. In *Proceedings on Privacy Enhancing Technologies*, 2022.

[34] C. Matte, M. Bielova, C. Santos, and M. Olejnik. Do cookie banners respect my choice?, measuring legal compliance of banners from iab europe's transparency and consent framework. *IEEE Symposium on Security and Privacy*, 2020.

[35] Clara Santos et al. Consent management platforms under the GDPR: cookies, dark patterns, and implications for privacy. *Journal of Data Protection & Privacy*, 4(2):157–178, 2021.

[36] Natalia Bielova, Cristiana Santos, and C. Matte. Web privacy: Dark patterns in cookie consent banners. *IEEE Security Privacy*, 2021.

[37] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 674–689, 2014.

[38] Lars Backstrom, Cynthia Dwork, Jon Kleinberg, Eva Tardos, and Michael Thorup. Group formation in large social networks: Membership, growth, and evolution. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 44–54, 2006.

[39] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. *IEEE Symposium on Security and Privacy*, pages 173–187, 2009.

[40] Bin Zhou, Jian Pei, and WoShun Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *ACM SIGKDD Explorations Newsletter*, 10(2):12–22, 2008.

[41] Shaoor Munir, Sandra Siby, Umar Iqbal, Steven Englehardt, Zubair Shafiq, and Carmela Troncoso. Cookiegraph: Understanding and detecting first-party tracking cookies. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 3490–3504, 2023.

[42] Lorrie Faith Cranor, Marc Langheinrich, Massimo Marchiori, Martin Presler-Marshall, and Joseph Reagle. The platform for privacy preferences 1.0 (p3p1.0) specification. In *W3C Recommendation*, 2002.

[43] Paul Ashley, Satoshi Hada, Günter Karjoth, Calvin Powers, Matthias Schunter, and Charles Western. Enterprise privacy authorization language (epal 1.2). In *IBM Research Report*, 2003.

[44] Florian Schaub, Rebecca Balebako, Adam L. Durity, and Lorrie Faith Cranor. A design space for effective privacy notices. In *Proceedings of the 11th Symposium On Usable Privacy and Security (SOUPS)*, 2015.

[45] Yongfeng Zhang, Xu Chen, Yiqun Liu, Min Zhang, Shaoping Ma, Lizi Liao, and Tat-Seng Chua. Collaborative knowledge base embedding for recommender systems. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 353–362, 2016.

[46] Mikhail J Atallah and Nicholas J Hopper. Erratum to: Privacy enhancing technologies. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages E1–E1. Springer, 2010.

[47] Ida Budin-Ljøsne, Jane Kaye, Deborah Beck, John E. Bentzen, Mats Hansson, Heidi M. Oien, Emmanuelle Rial-Sebbag, and Deborah Mascalzoni. Dynamic consent: A potential solution to some of the challenges of modern biomedical research. *BMC Medical Ethics*, 18(1):4, 2017.

[48] Jane Kaye, Naomi M. M. Whitley, Emmanuelle Rial-Sebbag, and Deborah Mascalzoni. Dynamic consent: A patient interface for twenty-first century research networks. *European Journal of Human Genetics*, 23(2):141–146, 2015.

[49] Wenfei Fan, Muyang Liu, Ping Lu, and Qiang Yin. Graph algorithms with partition transparency. *IEEE Transactions on Knowledge and Data Engineering*, 35(2):1554–1566, 2021.

[50] Australian Competition and Consumer Commission. *Digital Platforms Inquiry: Final Report*. Commonwealth of Australia, Canberra, Australia, 1st edition, 2019. ISBN 9781920702052. URL `https://www.accc.gov.au/publications/digital-platforms-inquiry-final-report`. ACCC Publication No. 06/19_1545.

[51] Ersin Dincelli, Emre Ozdemir, Lukasz Kusyk, Hazal Uzunkaya, Sarah Huo, and Pia Behmuaras. Countering digital procrastination: Dark design patterns and nudging strategies. In *29th Pacific Asia Conference on Information Systems, PACIS 2025, Kuala Lumpur, Malaysia, July 6-9, 2025*, 2025. URL `https://aisel.aisnet.org/pacis2025/hci/hci/10`.