# Free and Open-Source Software: Coordination and Competition

Robin Ng[†]

17th March 2024

**Abstract**

Free and Open-Source Software (FOSS) are developed by a community of developers led by a coordinator. Coordinators balance the following trade-off: (i) more developers improve FOSS' quality—a positive vertical differentiation effect; (ii) more developers lead to more diverse views, driving FOSS characteristics away from existing developers' preferences—a negative horizontal differentiation effect. To attract more developers, coordinators may improve their level of coordination, increasing the marginal vertical network effect, or adopt more permissive Open-Source licenses, increasing the marginal horizontal network effect. Permissive Open-Source licenses can intensify competition when FOSS compete with proprietary software, resulting in lower prices. However, permissive licenses may also reduces the incentives to coordinate FOSS, leading to lower quality FOSS that serve only a niche market. I study coordinators with different motivations: self-interested *Founders*, volunteering *Altruists*, and profit-driven *Managers*. Altruists and Managers fail to maximize total surplus, while Founders generate higher total surplus than Altruists.

**JEL:** D21, D26, L14, L17
**Keywords:** Open-Source Software, Network effects, Software Licensing

†Department of Economics, University of Mannheim; Mannheim Centre for Competition and Innovation.
**Contact:** hello@robinng.com
**Acknowledgements:** I am especially grateful to Paul Belleflamme, Greg Taylor, Johannes Johnen, and Rachel Tan for countless fruitful discussions. I also thank Carl-Christian Groh, Julius Goedde, Justin Johnson, François Maniquet, Martin Peitz, Erika Pini, and participants of the CORE Brown Bag, PSE EDP Jamboree, AMES2023, MaCCI Annual Conference for their comments.
**Fundng:** This project is realised with funding from: Fédération Wallonie-Bruxelles through the Action de recherche concertée grant 19/24-101; Belgian National Fund for Scientific Research through the Aspirant research fellowship FC46885; Deutsche Forschungsgemeinschaft (DFG) through the Collaborative Research Center Transregio (CRC TR) 224 Project B05.

# 1 Introduction

Free and Open-Source Software (FOSS) are a bedrock of the digital age. They range from prominent household names like Google's Android to niche and independently developed programs like Paint.NET. These software contribute to at least 2% of the global GDP (Blind & Schubert, 2023), and events in the FOSS community can have a wide-ranging impact. For example, the USA's 'Securing Open Source Software Act of 2023' was triggered by a FOSS vulnerability.[1] Although many firms and individuals adopt FOSS, few ever fund it.[2] Even the most prominent of FOSS receive funding below the USA poverty line.[3] Hence, developers of FOSS are effectively volunteers, having no obligation to develop any aspect of the software (Hars & Ou, 2002; Franke & Von Hippel, 2003; Shah, 2006). This community-driven nature of FOSS creates difficulties for those coordinating its development (Michlmayr, Hunt, & Probert, 2007).[4] Despite the lack of funding, and the challenges of coordinating FOSS development, many still choose to do so.

Prior research has chosen to abstract away from the decision to coordinate the development of FOSS. Instead, they focus on the private motives of developers contributing to a public good (Johnson, 2002, 2006; Casadesus-Masanell & Ghemawat, 2006; Athey & Ellison, 2014). I contribute to this body of work by considering the motives of coordinators and preferences of developers. By studying a theoretical model in which a FOSS is developed by a set of developers whose work is managed by a coordinator. I ask how FOSS development is coordinated and how FOSS compete with proprietary software.

Coordinators guide the development of FOSS. They act as an interface between users and developers, leading the community of developers by prioritising features and bugs to work on (Lerner & Tirole, 2002; Klug, Bogart, & Herbsleb, 2021).[5,6,7] Hence, higher levels of coordination can improve the quality of the FOSS as efforts by developers are more likely complementary. Conversely, less coordination leads to a lower quality FOSS as development efforts become more substitutable. I investigate three common motivations of coordinators (Lerner & Tirole, 2002): (i) **self-interested Founders** who are unsatisfied

[1] Other significant events include: the Heartbleed bug (John Biggs, 2014), malicious corruption by Marak Squires (Emma Roth, 2022), un-publication of leftpad (Chris Williams, 2016) to name a few.

[2] Firm adoption of FOSS is documented and encouraged by the Boston Consulting Group (2021). Denis Pushkarev (2023) and Marak Squires describe the funding challenges faced by developers (Emma Roth, 2022).

[3] André Staltz (2019) documents on the 58 most popular GitHub projects. The Apache Software Foundation (2023b) runs almost 400 FOSS projects on annual funding of USD 1 to 3 million.

[4] Chris Stokel-Walker (2014) and Denis Pushkarev (2023) document the problems they face in coordinating development.

[5] Raymond (1999) discusses the centrality of the coordinator's role and how he had to establish his competency to lead the FOSS Fetchmail, building relationships with users and developers.

[6] The Apache Software Foundation (2023a); The Document Foundation (2023); The Linux Foundation (2023) are examples of FOSS foundations which are governed by volunteers.

[7] Brian Proffitt (2016) describes how governance of FOSS is decided in a meritocratic manner on Red Hat's Blog.

with existing software, looking to maximise their own private utility (Klug et al., 2021); (ii) **volunteering Altruists** from the community who care about the outcomes of others (Raymond, 1999)[6]; (iii) **profit-driven Managers** who may provide paid features and services adjacent to the FOSS, maximising their own profits (Andersen-Gott, Ghinea, & Bygstad, 2012)[2,8]. To model this, I introduce coordinators who decide on the level of coordination the FOSS receives, focusing on self-interested Founders in the main model.

Developers choose when, what, and how they contribute to FOSS. Evidence shows developers contribute to FOSS because they are attracted to the high levels of customisability FOSS provide, only contributing to the development of FOSS features that directly benefit themselves (Raymond, 1999; Hars & Ou, 2002; Lerner & Tirole, 2002; Mustonen, 2003; Franke & Von Hippel, 2003; Shah, 2006). Selective contributions by developers mean the characteristics of a FOSS depend on who decides to contribute to it.

However, the ability of a developer to alter the characteristics of a FOSS depends on the governance of the FOSS. I consider the software license adopted by the FOSS as a proxy of its governance structure. More restrictive licenses, such as copyleft licenses, provides the coordinator with more control over the FOSS characteristics. This means the characteristics are less sensitive to the preferences of developers.[9] Conversely, permissive licenses, such as the MIT license, allow the characteristics of the FOSS to be more sensitive to developers' preferences.[10] I capture these effects in a model of product differentiation, allowing product characteristics to depend on which developers contribute to the FOSS and the software license adopted by the FOSS.

My main results discuss how Founders' trade-off quality and their own preferences over software characteristics, leading to an equilibrium level of coordination which depends on the FOSS license. I show that considering developers' preference and ability to alter the characteristics of FOSS induces a more competitive environment. When comparing across coordinators with different incentives, I show that Founders prefer a higher level of coordination than Altruists, and generate a higher total surplus.

Formally, I adopt a simple model of product differentiation à la Hotelling, modelling users with preferences over the characteristics of a software (Hotelling, 1929; d'Aspremont, Gabszewicz, & Thisse, 1979). I focus on the situation where a duopoly exists, users choose between paying a fee and using a proprietary software or paying nothing to use the FOSS. I assume that users are also developers, and where they choose to use the

[8]Prominent examples of firms with a profit-driven motive that actively manage their own FOSS include Microsoft, IBM, and Google.

[9]A FOSS with a copyleft license would require any modifications or extensions to the software to be made a FOSS. Meaning proprietary software cannot reuse code from copyleft software.

[10]Permissive licenses impose few restrictions on how the FOSS' source code is used or distributed. The MIT license is listed among the most popular open-source licenses by Open Source Initiative (2022) and GitHub (Balter (2015) blogs about this on GitHub.com).

FOSS, they contribute to its development.[11] Users face some mismatch cost if their preferred characteristics differ from the characteristics of the chosen software.

A proprietary software is located at one end of the Hotelling line, and has a fixed quality. Its price is set by a profit-maximising **proprietary Firm**.

A Founder's location is fixed at the opposite end of the Hotelling line. Founders decide on the level of coordination of the FOSS, facing the following trade-off: On the one hand, higher levels of coordination ensures that developers' efforts complement each other, leading to distinct contributions to the FOSS, improving the quality of the FOSS. Through a virtuous cycle of network effects, this attracts even more developers, further improving quality.

On the other hand, the characteristics of a FOSS are endogenous and depends on who develops it. Hence, the preferences of developers affect the location of the FOSS. Because the Founders' location is fixed at one end of the Hotelling line, every additional developer shifts the location of the FOSS away from the Founders' preference. This increases the mismatch cost for Founders, who may prefer lower levels of coordination to reduce their mismatch cost.

First, by considering how developers' incentives may influence the location of a FOSS, I show that FOSS generate more competitive effects than previously understood. When more developers contribute to the FOSS, the characteristics of the FOSS become closer to the proprietary software, leading to fiercer horizontal competition. This effect is in addition to previously studied scenarios where more users can improve product quality (Shy & Thisse, 1999; Fainmesser & Galeotti, 2020). Because more developers benefit the FOSS' competitive position in two dimensions, proprietary Firms have a stronger incentive to lower prices, increasing competition in the market. I discuss how government policies supporting the development of FOSS—even where the FOSS remains underutilised—can be beneficial as they drive competition (Madiega, 2020).[12]

Second, I show how the equilibrium level of coordination depends on the FOSS license. Restrictive licenses mean the FOSS location is not sensitive to the marginal developer. As developers are less able to influence the final software design, it is harder to encourage them to participate in the development of the FOSS. Fewer developers result in a FOSS of lower quality. Hence, when licenses are restrictive, Founders prefer distinct contributions to boost the quality of the FOSS. When a FOSS license is permissive, the location of the FOSS is more sensitive to the marginal developer, and additional developers increase the Founder's mismatch cost. As such, Founders may be deterred from managing the

[11] One way to think of this is that users are also contributing to the task of debugging the software (Raymond, 1999).

[12] See Sean Fleming (2021) and Benjamin Cedric Larsen (2022) for discussions on digital sovereignty.

FOSS, leading to a lower level of coordination. My model suggests that the prominence of permissive licenses, such as the MIT license, may explain why Founders decide to reduce, or even cease, their coordination efforts.[13]

Third, I conduct a similar analysis for FOSS coordinated by volunteering Altruists and profit-driven Managers. I find that Altruists behave similarly to Founders. However, to minimise mismatch cost, they prefer a lower level of coordination. Managers' level of coordination in binary, and depend on how competitive the proprietary software is. I show how this result can be applied to explain the rise of Google's Chrome.

Finally, I discuss the surplus generated by FOSS. I show that Founders generate a higher total surplus than Altruists. This result connects with anecdotal evidence on how many influential FOSS are coordinated by their Founders.[14] This also highlights the importance of directing FOSS funding to the correct coordinators, and questions how funding with stipulations of affiliation or requirements to adopt specific FOSS licenses can negatively affect the market. Furthermore, I discuss how Managers benefit from permissive licenses, which questions the motives of organisations which support the use of highly permissive licenses.

# 2 Related Literature

This paper studies the coordination of Free and Open-Source Software. Previous theoretical work has focused on the eccentricities of developers (Johnson, 2002, 2006; Casadesus-Masanell & Ghemawat, 2006; Athey & Ellison, 2014). My paper is perhaps most similar to Johnson (2002), as we consider how the community-driven nature of FOSS can influence the characteristics of the FOSS. I build upon this, taking into account the motives of coordinators, and study three types of coordinators: (i) the founder (Klug et al., 2021), (ii) members of the community who are recognised and appointed (Raymond, 1999), (iii) and firms.[15] See Lerner and Tirole (2002) for a substantive survey.

I extend the workhorse model of product differentiation provided in Hotelling (1929). This model has been used to study quality network effects, where more users can lead to better quality products (Shy & Thisse, 1999; Fainmesser & Galeotti, 2020). In addition to quality network effects, I introduce location network effects, where users can modify the location of the product. Using this model, I study the situation where a product's compatibility with users can depend on the preferences of other users. Doing so allows me to capture empirical evidence on the motives of developers who choose when, what, and

[13] For example, the malicious corruption of color.js and faker.js by Marak Squires (Emma Roth, 2022) and the un-publication of leftpad by Chris Stokel-Walker (2014).

[14] Examples include core-js (Denis Pushkarev, 2023) and leftpad (Chris Williams, 2016). Also, see the xkcd comic on dependency (https://xkcd.com/2347/).

[15] Prominent examples of firms that actively manage their own FOSS include Microsoft, IBM, and Google.

how they contribute to FOSS (Fielding, 1999; Hars & Ou, 2002; Franke & Von Hippel, 2003; Shah, 2006).

By capturing the motives of developers, this paper relates to the literature on crowd-sourced innovation and competition. I show that a FOSS with a small market share can have large competitive effects, and this is driven by the proprietary Firm preventing the development of the FOSS by lowering prices. My result is similar to Boone (2001). Through a different mechanism, Boone (2001) shows that market concentration can drive innovation by the incumbent, and does not imply a lack of competition.

There is little theoretical work discussing open-source licenses. Most papers avoid this discussion or discuss their results in the context of a specific license. Lerner and Tirole (2005b) and Gaudeul (2005) discuss simple models of license choice. Gaudeul (2008) assumes the selection of a given binary license, either permissive or restrictive, and discuss the implications of using either license. August, Chen, and Zhu (2021) argues that restrictive licenses are analogous to larger contributor network effects, showing how restrictive licenses can be detrimental.

My model also takes the license choice as given, but allows for a potential continuum of license designs. I study how coordinators select the level of coordination, and hence contributor network effects. In comparative statics, I show how the choice of FOSS license can affect its coordination and ability to compete with proprietary products.

In Section 3, I describe the setting in detail, discussing my main results in Section 4, and surplus implications in Section 5. In Section 6, I provide a series of extensions, and Section 7 concludes. I delegate proofs to Appendix A and B.

# 3 Model

I begin with the construction of a standard Hotelling model. I then introduce the concept of a FOSS, which accounts for network effects that simultaneously influence both the quality and location of the FOSS.

**User.** There exists a unit mass of users. Users have heterogeneous preferences for product characteristics, $x$, and these are uniformly distributed, $x \sim U(0,1)$, along a Hotelling line. Users may choose between consuming the FOSS or an existing proprietary software, $i \in \{o, p\}$ respectively. Let $\bar{x}$ be the location of the user indifferent between consuming the FOSS or the proprietary software. The utility users receive from each software is evaluated as follows: $u_i = v_i - p_i - t|L_i - x|$. $v_i$ represents the quality of the software, $p_i$ the price, $L_i$ the location of the software, and $t > 0$ the cost associated with a poor match ('mismatch cost').

**Firm.** A proprietary **Firm** produces an existing proprietary software. This software is located at one end of the Hotelling line, fixing $L_p = 0$. The Firm is endowed with a software of quality $v_p$, which is large enough to ensure that the market is covered in equilibrium. The Firm sets a price, $p_p$, that allows it to maximise its profits, $\pi_p = p_p \cdot D_p$, where $D_p$ is demand for the Firm's software. In this setting, $D_p = max\{\bar{x}, 1\}$, $\bar{x}$ being the indifferent user.

By fixing the proprietary software's location at 0, I study the situation where the Firm is unable to change the characteristics of its software. Seemingly a strict assumption, this is rooted in the notion that the Firm is an incumbent which has already decided what features its software is going to possess.[16] Moreover, this simplification allows me to focus on addressing the development of FOSS.

Thus far, I have described a standard Hotelling model with a fixed location, where a proprietary Firm only makes a pricing decision. To this model, I introduce a Free and Open-Source Software whose quality and location are influenced by the network of developers contributing to it.

**Developers.** There exists a unit mass of developers. Developers have heterogeneous preferences for what product characteristics they wish to develop, $o$, and these are uniformly distributed, $o \sim U(0, 1)$, along the same Hotelling line as users. They receive a utility of $w_o = s_o - k|L_o - o|$. $s_o$ captures the benefits to a developer and this depends on the number of users of the FOSS, $s_o = \beta \cdot D_o$, where $D_o$ is the mass of users using the FOSS and $\beta$ captures the prominence each additional user brings to the software. $k > 0$ captures the cost associated with developing features for a software that does not perfectly align with the developers preferences. Developers may either choose to contribute to the software, or some outside option normalised to 0.[17]

Modelling developers this way captures how the popularity of a software allows the skills of individual developers to be recognised. This benefits the developers if they were motivated intrinsically to build something useful, or extrinsically, translating to job opportunities. Moreover, the mismatch cost here captures the preference to work on features of a software that may interest or benefit developers directly, and much less on other features.

**Free and Open-Source Software.** To capture the characteristics of a FOSS, I introduce a model of endogenous quality (vertical differentiation) and location (horizontal differentiation), both of which depend on the number of users. I dub these effects 'quality' and 'location' network effects respectively.[18] By definition, a FOSS is free to use, and

[16] For example, it is public knowledge when Microsoft releases a feature, and what features are under development (Microsoft, 2023a, 2023b).

[17] Simply doing nothing has no cost.

[18] Network effects arise when each user's utility is affected by the number of other users. My main model captures 'direct' network effects. In the general setting where users and developers are mutually

hence $p_o$, the price of the FOSS, is 0.[19]

The quality of the FOSS depends on the number of developers, $v_o = v_c + \gamma \cdot D_o$. Here, $\gamma \in [0, 1]$ captures the probability that each contribution is distinct. Because developers are free to select the type of contributions they make, there is always a possibility that multiple contributions overlap.[20] The quality of the FOSS when the coordinator is its sole developer is represented by $v_c$. This captures that some FOSS may exist without additional developers, and thus do not benefit from network effect. I normalise $v_c = 0$ to reduce parameters.[21]

The location of the FOSS depends on the preferences of its developers, $L_o = 1 - l \cdot D_o$, $l \in (0, 1)$ is a parameter that regulates the FOSS' location, and $D_o = 1 - \bar{x}$ is the demand the FOSS faces, $\bar{x}$ being the marginal developer. In particular, $l \to 0$ suggests that the location of the software is insensitive to the marginal developer, and $l \to 1$ implies that the location is very sensitive to the marginal developer. Mathematically, when more developers contribute to the development of the FOSS, the location of the FOSS shifts closer to the proprietary software. I interpret $l$ as the permissiveness of the license adopted by the FOSS, and suppose that $l$ is fixed ex-ante.[22]

A permissive license guarantees the right to use modify, and redistribute the underlying source code. More restrictive licenses may grant limited rights, disallowing the software to be used in specific situations, for example for profit or in programs that may harm others. The most restrictive of licenses would be a closed source software where the underlying source code is public but there is no right of use, or a proprietary license where no code is made public. Hence, permissive licenses encourages users to suggest modifications to the coordinator, and make a coordinator more likely to accept changes.[23] This way, I interpret $l$ as the permissiveness of a license, which relates to the ability of developers to influence the characteristics and location of the FOSS.

**Coordinator.** Actively developed Free and Open-Source Software are always coordinated by someone. The coordinator chooses the probability that each contribution is distinct, $\gamma \in [0, 1]$. This proxies the coordination level of the FOSS, such that a larger $\gamma$ reflects how the efforts of developers complement each other. In the main model, I dis-

exclusive, network effects are 'indirect'. See Belleflamme and Peitz (2015) for details.

[19] FOSS are free to use because anyone is able to download it source code to build and run the software.

[20] And $(1 - \gamma)$ the probability that the contributions overlap.

[21] More generally, the reader may choose to interpret $v_p > 0$ as the difference between the quality of the proprietary software and the FOSS without network effects. Notice that $v_p \leq 0$ implies that the Firm is inactive. See Etzion and Pang (2014) for details.

[22] The reader may also choose to interpret $l$ as the hierarchical structure of the FOSS, or the inverse of the degree of control that coordinators have over the direction of the FOSS.

[23] When the FOSS has a permissive license and the coordinator chooses not to accept a change, other developers can simply 'fork' or spin-off the project. Denis Pushkarev (2023), the founder of core-js, a standard library for JavaScript, describes this in his regular update to the community.

cuss a FOSS which is coordinated by its self-interested **Founder**. Founders are motivated to develop a FOSS because they face a huge mismatch cost from using the proprietary software. Hence, I assume that Founders are located at 1. Moreover, Founders develop FOSS to fix problems of their own. I model Founders as self-interested, maximising their own utility, $\pi_o = \gamma \cdot D_o - t(1 - L_o)$, where $\gamma$ is the level of coordination, $D_o$ the mass of FOSS developers, $t > 0$ a mismatch cost. Since the Founder is located at 1, $1 - L_o = 1 - (1 - l \cdot D_o) = l \cdot D_o$, where $l \in (0, 1)$ is a parameter that regulates the FOSS' location. This utility function is identical to the utility of function of users, but is specific to a user located at 1.

For simplicity, I shall assume that all users are developers, collapsing both groups into 'users'. This is equivalent to saying that individuals may choose to participate in the development of, and use, a FOSS, or instead simply use a proprietary software. Mathematically, this means setting $\beta = \gamma$, $o = x$, and $k = t$. This reduces the number of actors, and variables, in the main model and allows me to focus on the decision of the coordinator.

The sequence of events follows: The coordinator chooses the probability that each contribution is distinct, $\gamma$, the Firm then sets the price, $p_p$, and users finally decide between the proprietary software or the FOSS. This captures the notion that it takes time to develop a software and the fluidity of software prices.

This concludes the description of the model environment.

**Restrictions.** I look for a subgame perfect equilibrium. To characterise this equilibrium, I introduce the following restrictions as equilibrium selection assumptions. These restrictions mirror standard restrictions in Hotelling models.

**Restriction 1.** *The market is covered.*

**Restriction 2.** *Both the Firm and the FOSS are active in the market.*

Combined, these restrictions allow me to focus on the scenario where the development of a FOSS creates competitive pressures for the Firm. This is the scenario of interest as the evidence suggests that many FOSS are developed as a response to the needs of users which cannot be met by existing software - meaning that in many cases, there exists a proprietary software that serves as a less than perfect substitute. This allows me to study a rich environment, and explain many realistic competitive environments.

# 4 Coordinating Free and Open-Source Software

Before considering the decisions of coordinators, I determine the choices made by users and the Firm. Their decisions are the best response functions to the decision of other

users and the coordinator, and do not depend on the motives of the coordinator. While the quality of the proprietary software and the mismatch cost affect the equilibrium, these are economically understood. Instead, I focus my discussion on the novel effects that arise from the coordination and license of FOSS.

First, I determine the users' choice of software: the FOSS or the proprietary software. In this setting, there exists a user who is indifferent between the FOSS and the proprietary software,

$$u_o(\bar{x}) = u_p(\bar{x})$$
$$\gamma \cdot D_o - p_o - t|L_o - \bar{x}| = v_p - p_p - t\bar{x}$$

where $D_o = (1 - \bar{x})$, $L_o = (1 - l(1 - \bar{x}))$, and $L_o \geq \bar{x}$ because $\bar{x}$ is the indifferent user.

$$\gamma \cdot (1 - \bar{x}) - p_o - t(1 - l)(1 - \bar{x}) = v_p - p_p - t\bar{x}$$
$$\bar{x} = \frac{v_p - p_p + p_o - \gamma + t(1 - l)}{t(2 - l) - \gamma} \tag{1}$$

Intuitively, higher levels of coordination increases the marginal network effect ($\gamma \uparrow$), improving the quality of the FOSS. Assuming prices to be 0, and hence ignoring the strategic pricing decision of the Firm, this makes the FOSS more attractive to users.

To understand the role FOSS license plays, consider that more permissive licenses ($l \uparrow$) give more rights and freedom to other developers to modify and use the FOSS code. Conversely, restrictive licenses prevent the use of the underlying code, the most extreme example being closed proprietary software, where only the binary file is shared, removing the ability of others to view (and hence use) the underlying code. Therefore permissive software licenses mean the characteristics of the FOSS are more sensitive to the marginal user. When a FOSS license is permissive, the marginal user faces a lower mismatch cost when choosing the FOSS. Hence, by ignoring the strategic pricing decisions of the Firm and assuming the price of FOSS to be 0, permissive licenses also increase the FOSS' appeal to users.

Second, I consider the decision of the Firm. The Firm chooses to lower prices when the FOSS has a higher level of coordination or a more permissive license. First, a higher level of coordination increases the quality of the FOSS, which leads to a virtuous cycle of network effects where the higher quality FOSS attracts even more users. Firms hence faces stronger quality-based (vertical) competition. Second, more permissive licenses mean the characteristics of the FOSS are more sensitive to the preferences of the marginal user. This means as more users use the FOSS, the location of the FOSS becomes closer to the proprietary software. This makes the characteristics of both products more similar,

increasing location-based (horizontal) competition.

**Lemma 1.** *The equilibrium price set by the proprietary Firm strictly increases in its quality and the price of the FOSS, and strictly decreases in the level of coordination of the FOSS and permissiveness of the FOSS license.*

$$p_p^* = \frac{v_p + p_o - \gamma + t(1-l)}{2} \tag{2}$$

*See A.1 for proof.*

Because FOSS are free to use, I impose that $p_o = 0$. In equilibrium, when accounting for the price of the proprietary software, (1) becomes

$$\bar{x} = \frac{v_p - \gamma + t(1-l)}{2(t(2-l) - \gamma)}$$

and the first derivative with respect to $\gamma$ and $l$ are positive if and only if $v_p > t$.[24] Therefore, firms with a sufficiently high quality software are able to capture a larger share of the market by setting lower prices even when FOSS have a higher level of coordination or more permissive licenses.

Interestingly this means that setting a lower price allows the Firm to recapture demand and counter competitive forces generated by the FOSS in two ways. First, by decreasing the size of the FOSS network, the quality of the FOSS decreases, reducing quality-based competition. Second, the Firm recaptures users with less extreme preferences, this causes only users with more extreme preferences (closer to 1) to remain in the FOSS network, increasing the horizontal differentiation between the two software, reducing location-based competition.

The dual incentive for the Firm to compete on prices suggests that the existence of FOSS is able to generate a larger competitive effect than previously understood. Because standard models of competition do not account for how FOSS' characteristics depend on developers' preferences, they do not account for the additional location competitive effect. Lemma 1 and $\bar{x}$ together show that firms have an additional incentive to lower prices when competing with FOSS which have more permissive licenses. Hence, by accounting for this location competitive effect, I show that more permissive FOSS licenses can reinforce competition.

Despite gaining a larger share of the market, the proprietary software is less profitable when competing against a FOSS with a higher level of coordination or a more permissive

[24] Consider the incentives of a FOSS coordinator. In the case of Founders or Altruists, they provide a service for free, and many have argued that no money should change hands. See Footnote 12 and 19. When considering profit-driven Managers, negative pricing cannot be profitable.

license. The negative effect on profit from the fall in prices dominates any gains in profit from a larger market share.

**Corollary 1.** *Firms' profits strictly decreases in the level of coordination of the FOSS and permissiveness of the FOSS license.*

$$\pi_p = \frac{[v_p + p_o - \gamma + t(1-l)]^2}{4(t(2-l)-\gamma)}$$

*See A.1 for proof.*

Corollary 1 and $\bar{x}$ shows how Firms profits decrease when coordination levels are higher, yet Firms are able to capture a larger market share. This suggests that even niche FOSS that serve small communities can contribute to a more competitive environment. This result has similar implications as Boone (2001), highlighting the failures of using market concentration as a measure of market competitiveness.

Having discussed the decisions of users and Firms, I now turn my attention to self-interested Founders.

## 4.1 Self-Interested Founder

As a baseline, I consider the incentives of FOSS **Founders**. As suggested by many, Founders of FOSS are usually motivated by some degree of self-interest, looking to resolve a problems that they face (Lerner & Tirole, 2002; Hars & Ou, 2002; Franke & Von Hippel, 2003; Klug et al., 2021). I take an extreme view where Founders are completely self-interested, seeking to maximise their own utility.

Recall that Founders utility is

$$\begin{aligned}\pi_o &= \gamma \cdot D_o - t(l \cdot D_o) \\ &= \frac{(\gamma - tl)(t(3-l) - v_p - \gamma)}{2(t(2-l)-\gamma)}\end{aligned}$$

and the FOSS is free to use, $p_o = 0$. Founders decide on the probability that contributions are distinct, $\gamma$, which proxies the level of coordination of the FOSS. This provides the following first-order condition:

$$\frac{\partial \pi_o}{\partial \gamma} = \underbrace{\frac{(\gamma - tl)(t(3-l) - v_p - \gamma)}{2(t(2-l)-\gamma)^2} + \frac{t(3-l) - v_p - \gamma}{2(t(2-l)-\gamma)}}_{>0} \underbrace{- \frac{\gamma - tl}{2(t(2-l)-\gamma)}}_{<0}.$$

The first-order condition shows Founders trade-off. On the one hand, they have an incentive to grow the network of users. By leveraging the contributions of additional users, Founders can improve the quality of the FOSS, and by extension, their own utility.

Hence, Founders prefer that contributions are somewhat distinct, $\gamma > 0$.

However, as more users join the network, Founders faces two opposing forces. First, the characteristics of the FOSS change. This change results in a software that is less compatible with Founders needs and preferences, shifting the location of the software away from Founders. Under these circumstances, Founders have less incentive to ensure that contributions to the development of the FOSS are distinct, as doing so incurs a higher mismatch cost, harming themselves. Second, Firms lower prices in response to fiercer quality competition, allowing the proprietary software to become more attractive to users. As a result, more users choose to use the proprietary software, and the FOSS benefits from less network effects. In this way, Founders becomes a victim of their own success as they now pose a threat to Firms.

Using this trade-off, I find there exists a unique subgame perfect Nash equilibrium that satisfies Restrictions 1 and 2.

**Proposition 1.** *Founders are active only if the proprietary software is of a low quality,* $(v_p < t(3-2l))$. *Their level of coordination is weakly decreasing in the permissiveness of the FOSS license* $(l \uparrow)$*, and the quality of the proprietary software, and strictly decreasing if contributions are not perfectly distinct* $(\gamma^* < 1)$.

$$\gamma^* = min\{t(2-l) - \sqrt{2t(v_p - t)(1-l)}, 1\}$$

*with* $\gamma^* = 1 \iff 3t + \frac{tl^2}{2(1-l)} + \frac{1}{2t(1-l)} - \frac{2-l}{1-l} \geq v_p$ *and* $t > \frac{1}{2-l}$.

*See A.2 for proof.*

Proposition 1 reiterates how more permissive licenses can make Founders worse-off. Because more permissive licenses allow other developers to have a greater influence over the characteristics of the FOSS, the location of the software is more sensitive to the preferences of the marginal user. Hence, more permissive licenses increase the mismatch cost for Founders.

In response, Founders of permissive FOSS hence choose lower levels of coordination, allowing overlapping contributions to the FOSS. Doing so has a direct effect of reducing the quality of the FOSS. However, a lower quality FOSS attracts less users, which indirectly reduces the mismatch cost of Founders. The lower quality dominates the recovered mismatch cost, making Founders worse-off.

One way of understanding such licenses is in the ability to control the future of the underlying code. This is not dissimilar to our understanding of Intellectual Property Rights (IPR), where a larger $l$ captures limited IPRs for the rights holder. This way, Proposition 1 connects well to prior research on how IPRs can lead to more investment

and innovation (Bloxam, 1957; Besley, 1995; Chen & Puttitanun, 2005; Moser, 2005; Qian, 2007).

Software licenses are usually selected at the onset (Vendome et al., 2015b).[25] Because of the collaborative nature of FOSS it is often not possible to modify a license once it has been adopted (Gamblin, 2021).[26] This can lead to a license "lock-in" effect. Anecdotal evidence suggests that Founders are encouraged to, and often select permissive FOSS licenses—such as the MIT license.[10] However, this has been the subject of a long-standing debate.

Supporters of permissive licenses argue that permissive licenses ease the rules of participation in the FOSS community: (i) Coordinators minimise their legal liability; (ii) Users face no legal risk for using the FOSS; (iii) Developers find it easier to contribute to the FOSS (Gamblin, 2021).[27] Permissive licenses thus encourage more users to join the FOSS community, allowing Founders to depend on network effects to grow the FOSS (Gamblin, 2021). They also argue that permissive licenses are in the spirit of FOSS, allowing anyone (and everyone) access (and permission) to use and modify the source code.

On the other hand, detractors claim that permissive licenses go against the spirit of liberty and digital sovereignty—the ideological view that the inability to pay for a proprietary software should not exclude individuals from participating in the digital age.[12,27] They worry that overly permissive licenses allow corporations to take advantage of FOSS developers, repacking FOSS into proprietary software and profiteering from a public good.[28] Still others warn that permissive licenses make it difficult for developers and coordinators to monetise their work, which can lead to unsustainable software development.

While I do not address the legal or ideological arguments made by these groups, I contribute to the discussion by understanding the economic implications of more permissive licenses.

Corollary 2 contributes to this discussion by showing how permissive licenses can lead to the underdevelopment of FOSS. Licenses need to provide Founders with sufficient control over the use and development of the FOSS. This level of control depends on $v_p$ and $t$. The differences in market conditions across each software market can thus explain the variety of open-source licenses in active use, and the continued challenges in designing

[25] Vendome et al. (2015a, 2015b) also comment that where a license change occurs, a more permissive license is subsequently adopted and this is often attributed to developers or funding requests, or to allow for commercialisation.

[26] Stackexchange (2015) provides a detailed discussion regarding the problems of subsequently changing FOSS licenses.

[27] Nicolas Suzor (2013) describes on the website opensource.com.

[28] Paint.NET was released in 2004 with the MIT license, developer Rick Brewster decided to adopt a stricter license to prevent others from profiteering from the software (Brewster (2009) explains in his community update).

and choosing boilerplate open-source licenses (Lerner & Tirole, 2002; Subramaniam, Sen, & Nelson, 2009).[29,30]

**Corollary 2.** *Founders only coordinate FOSS when the FOSS license is not too permissive. The upper bound on the permissiveness of the FOSS license strictly decreases in the quality of the proprietary software ($v_p \uparrow$), and strictly increases in the strength of user preferences ($t \uparrow$).*

*Mathematically,* $\gamma^* > 0 \iff l < \bar{l} = \frac{3t - v_p}{2t}$.

*See A.2 for proof.*

The intuition follows: when users' preferences are weaker (small $t$), they only face a small mismatch cost when using products with characteristics further from their preferences. This means the FOSS is less able to use location to improve the utility of users. As a result, the FOSS' advantage from location network effects diminishes. Hence, Founders are only active when licenses are more restrictive.

While permissive licenses may help avoid legal issues, they diminish the willingness of Founders to manage and continue the development of FOSS. This is not easy to observe in the data, but many prominent anecdotes where Founders have stopped developing FOSS exist. For instance, leftpad, colors.js and faker.js.[1] Others, such as Paint.NET, have adopted more restrictive licenses when market conditions evolved.[28] Corollary 2 hence provides a non-monetary mechanism in support of those who believe permissive software licenses may lead to unsustainable software development. My result questions the default recommendation of permissive licenses, such as the MIT license. Instead, FOSS licenses should be tailored for each software, not unlike IPRs, and no license should be promoted generically above another without consideration for the specific software market.

**Corollary 3.** *When a FOSS license is more permissive, the quality of the FOSS decreases, and the FOSS serves a smaller share of the market.*

*Proof.*

[29] Almeida, Murphy, Wilson, and Hoye (2017) shows developers have a hard time comparing licenses, turning to tools such as choosealicense.com and license.md.

[30] See also Janis Lesinski (2020) for problems with the MIT license.

In equilibrium, $v_o$ and (1) become

$$\begin{aligned} v_o &= \gamma(1-\bar{x}) \\ \bar{x} &= \frac{v_p - \gamma^* + t(1-l)}{2(t(2-l)-\gamma^*)} \\ \frac{\partial \bar{x}}{\partial l} &= \frac{t(v_p-t)^2}{2(2t(v_p-t)(1-l))^{\frac{3}{2}}} \\ \frac{\partial v_o}{\partial l} &= \frac{\partial \gamma}{\partial l}(1-\bar{x}) - \frac{\partial \bar{x}}{\partial l}\gamma. \end{aligned}$$

Notice that $v_p > t$ for a real solution to $\gamma^*$. Therefore $\frac{\partial \bar{x}}{\partial l} > 0$. Since $\frac{\partial \gamma}{\partial l} \leq 0$, $\frac{\partial v_o}{\partial l} < 0$. □

The negative impact permissive licenses have on the level of coordination (Proposition 1) and the price of the proprietary software (Lemma 1) can explain why many FOSS remain of a low quality, serving only niche markets and never becoming mainstream (Corollary 3) (Lakhani & Hippel, 2004; Lerner & Tirole, 2005a; Robles, Steinmacher, Adams, & Treude, 2019). First, it is not beneficial for Founders to over-invest in the development of the FOSS. Despite the virtuous cycle of network effects, where quality leads to more users, and more users lead to a software with mass market appeal, Founders suffer from a growing mismatch of preferences. Second, the proprietary Firm respond by becoming more price competitive, reducing the incentive for users to use the FOSS.

Hence, my model provides an explanation for the 'pet project' mentality found in anecdotal evidence.[31] That is to say, self-interested Founders may limit the quality and scope of the software despite the potential societal benefits, leading to the many FOSS which serve only niche markets or just a few individuals.

To summarise, the main model studies how the design of FOSS licenses influence the development and competitiveness of the FOSS. I show that self-interested Founders only manage a FOSS if they retain sufficient control of the software. When a FOSS license is more restrictive, Founders prefer that each contribution to the FOSS is distinct. However, if a more permissive license is adopted, Founders may prefer overlapping contributions, causing the FOSS to be of a lower quality, and attract fewer users. When licenses are permissive, despite being of a lower quality, the presence of a FOSS still leads proprietary Firms to lower prices, creating a more competitive environment.

## 4.2 Volunteering Altruist

The literature suggests that social preferences have been noted as a motivation for contributing to and managing a FOSS. Hars and Ou (2002) suggest that developers are often motivated by building a sense of community. Whereas Lerner and Tirole (2002) state

[31] Adem Ait Fonollà (2022) documents that most projects on GitHub are abandoned within 5 years.

that altruism is one motivation often reported. Moreover, many large FOSS communities are user managed: through a meritocratic process, users are recognised and rise in the hierarchy (Raymond, 1999).[6] Arguably, this implies that coordinators (as an individual or a group) are motivated by some sense of altruism. Volunteering **Altruists** are also more likely to be open to changing the characteristics of a FOSS to fit the needs of the users of the software. To model such Altruists, I consider that a coordinator might be motivated by a sense of community, working towards maximising the value of the FOSS for its users.

This model follows directly from the base setting, with the following modification: instead of a self-interested Founder, the coordinator is a volunteering Altruist and seeks to maximise the value of the FOSS network. The FOSS also continues to be free to use, $p_o = 0$. This means Altruists are concerned with generating the largest possible benefit for their members,

$$\begin{aligned}\pi_o^A &= \int_{\bar{x}}^1 u_o(x)dx \\ &= \int_{\bar{x}}^1 v_o - t|L_o - x|dx \\ &= \int_{\bar{x}}^1 \gamma(1-\bar{x}) - t|1 - l(1-\bar{x}) - x|dx.\end{aligned}$$

Taking into account the equilibrium decisions of users, (1), and Firms, (2),

$$\pi_o^A = \frac{(2\gamma - t(1-2l+2l^2))(v_p + \gamma - t(3-l))^2}{8(t(2-l)-\gamma)^2}.$$

Where coordinators are Altruists, I find that there exists a unique subgame perfect Nash equilibrium, satisfying Restrictions 1 and 2.

**Proposition 2.** *Altruists are active only if the proprietary software is of a low quality* ($v_p < t\frac{5-2l^2}{2}$). *Their level of coordination is weakly decreasing in the quality of the proprietary software, and strictly decreasing if contributions are not perfectly distinct* ($\gamma^A < 1$).

$$\gamma^A = min\left\{\frac{t(3-2l) + v_p - \sqrt{(v_p - t)(11t - 8tl^2 + v_p)}}{2}, 1\right\}$$

*with* $\gamma^A = 1 \iff v_p \in (t, \frac{1+t^2(5-3l-l^2)-t(3-2l)}{1+t(1+l-2l^2)}]$, *and* $t > \frac{1}{2-l}$.

*See A.3 for proof.*

To see when Altruists are active, first observe that $v_p < t\frac{5-2l^2}{2} \iff l < \sqrt{\frac{5t-2v_p}{2t}}$. Hence, Altruists face qualitatively similar trade-offs as Founders, and the intuition is the same as Corollary 2.

However, unlike self-interested Founders, Altruists level of coordination is not monotone in license permissiveness. For some intuition, consider the adoption of a more permissive license, the characteristics of the FOSS become further from 1. This increases the mismatch cost of those closer to 1. However, more users also improve the quality of the FOSS. This vertical network effect attracts even more users to the FOSS, and the new marginal user has preferences that are more different from the FOSS than the previous marginal user. As a result, the overall mismatch cost of FOSS users increases. Suppose that Altruists increase their coordination levels in response, this can improve the quality of the FOSS, but also further increase the mismatch cost to all users. Alternatively, Altruists may choose to decrease their level of coordination, reducing quality. This attracts less users, but further serves to reduce total mismatch cost. Because the total change in mismatch cost depends on the type of software license, the level of coordination is not monotone in the type of license.

**Corollary 4.** *Founders prefer more distinct contributions than Altruists. Hence,* $\gamma^* > \gamma^A$ *when* $\gamma^* \neq 1$ *and* $\gamma^A \neq 1$.

*See A.3 for proof.*

Corollary 4 compares the degree of distinct contributions Founders and Altruists prefer. To understand Corollary 4, notice that the key difference between FOSS coordinated by Founders and Altruists is the objective function of the coordinator. Founders are concerned with their own mismatch cost, whereas Altruists consider the mismatch cost of all FOSS users. This means Altruists are less willing to select a higher level of coordination, because this increases the costs of many others, while Founders only consider their personal costs.

For intuition, suppose that a coordinator selects a lower level of coordination ($\gamma \downarrow$). This decreases the quality of the FOSS. By extension, reducing the number of FOSS users, and shifting the location of the FOSS closer to 1. As a result, the mismatch cost of the extreme user, Founders located at 1, decreases. Additionally, the mismatch cost of the marginal user decreases. Because the Altruist concerns itself with all users of the FOSS, they consider this additional effect. Thus, Altruists have a larger benefit from selecting a lower $\gamma$ than Founders.

## 4.3 Profit-Driven Manager

Many prominent coordinators, such as Google and IBM, are driven by a profit motive. Lerner and Tirole (2002) and Andersen-Gott et al. (2012) also discuss how companies may adopt a code-release strategy, allowing the community to further develop the software, while providing features and services adjacent to the FOSS for profit. This setting is analogous to one where some of the software's code is open-source and other (non-employee)

developers may create plug-ins or extensions to the software. Although companies may allow this, they may still choose to moderate the type of plug-ins and extensions readily available. Companies then earns profit from the sale of data or other auxiliary services. A prime example of this is Google's Chrome web browser.

To model profit-driven **Managers**, I deviate from the main model and allow Managers to set prices and the level of coordination with profit in mind. Profit-driven Managers maximise the following $\pi_o^M = p_o \cdot D_o$, where $D_o = 1 - \bar{x}$ is the demand of the FOSS, and $p_o$ the price set by Managers. The timing of the game follows: (i) the Manager selects the probability that contributions are distinct, $\gamma$; (ii) the Manager and the proprietary Firm simultaneously select prices, $p_o$ and $p_p$ respectively; (iii) users decide between the FOSS or the proprietary software.

Taking into account the equilibrium decisions of users, (1),

$$\pi_o^M = \frac{p_o(p_p + t - v_p - p_o)}{t(2-l) - \gamma}$$

and it is immediate that profits are concave in price, and the optimal pricing strategy is $p_o^M = \frac{p_p + t - v_p}{2}$. Accounting for the pricing strategy of Firm, (2),

$$p_o^M = \frac{t(3-l) - \gamma - v_p}{3} \tag{3}$$

$$p_p^M = \frac{v_p - 2\gamma + t(3-2l)}{3} \tag{4}$$

Higher levels of coordination should lead to higher quality products. Despite this, Managers choose to lower prices when selecting a higher level of coordination. This is because lower prices have the additional effect of attracting more users. Hence, lower prices complement the higher level of coordination, building upon the virtuous cycle of network effects, to further improve the FOSS' quality. Together, equations (3) and (4) show how higher levels of coordination can lead to a more competitive environment, as Firms and Managers engage in fiercer price competition, lowering prices.

Taking into account equilibrium prices, (1) becomes

$$\bar{x} = \frac{t(3-2l) + v_p - 2\gamma}{3(t(2-l) - \gamma)}$$

$$\frac{\partial \bar{x}}{\partial \gamma} = \frac{v_p - t}{3(t(2-l) - \gamma)^2} \tag{5}$$

$$\frac{\partial \bar{x}}{\partial l} = \frac{t(v_p - t)}{3(t(2-l) - \gamma)^2} \tag{6}$$

When the quality of the proprietary software is sufficiently high ($v_p > t$), Firms have a

dual incentive to compete on prices. First, setting a lower price allows it to capture a larger share of the market. Second, increasing the market share of the proprietary software lowers the quality of the FOSS as less developers contribute to it. Hence, in response to a larger $\gamma$, Firms decrease their prices to capture a larger share of the market.

However, when the quality of the proprietary software is low ($t > v_p$), the mismatch cost faced by users is too large; this diminishes Firms' ability to use prices to gain market share. Hence, when the FOSS' level of coordination is increases, Firms instead lose consumers to the FOSS.

**Proposition 3.** *Managers are active only if the FOSS has a sufficiently permissive license ($l \geq \frac{3t-2}{2t}$). When active, profit-driven Managers choose between binary levels of coordination:*

- *They prefer distinct contributions, $\gamma^M = 1$, when the proprietary software is of a low quality, $v_p \leq t$.*
- *They prefer the most overlap, $\gamma^M = max\{t, t(1.5 - l)\}$, when the proprietary software has a high quality, $v_p > t$.*

*They prefer distinct contributions ($\gamma^M = 1$) when Firms have a low quality ($v_p < t$). When Firms have a high quality ($v_p \geq t$), Managers choose the lowest level of coordination ($\gamma^M = max\{t, t(1.5 - l)\}$) for which users continue to use the FOSS.*

*See A.4 for proof.*

Proposition 3 shows how Managers competing with higher quality proprietary software can take advantage of the permissiveness of FOSS to lower their level of coordination. Notice that when the FOSS license is restrictive, then $\gamma^M = t(1.5 - l)$, but as $l$ increases, or FOSS adopt more permissive licenses, $\gamma^M$ becomes smaller until $\gamma^M = t$. This result suggests that Managers adopting an open-source approach to software development can choose lower levels of coordination, which may make it easier for them to coordinate a FOSS. Moreover, despite a lower level of coordination, the coordination of the FOSS is still profitable.

Proposition 3 also shows how Managers of FOSS prefer distinct contributions when they face low quality proprietary competitors. This is driven by the gains in revenue from a larger user base (equation (5)), which dominates any loss that arises from a lower per-unit price (equation (4)).

These results connect with recommendations that companies should adopt open-source business models.[32] Anecdotal evidence shows how Microsoft, which once called open-source 'a cancer', has recanted on this position, and is now the largest contributor to the

[32] Consulting companies such as Boston Consulting Group (2021) promote that companies rely on the open-source community for software development.

development of FOSS.[33]

To further illustrate Proposition 3, consider the competition between internet browsers as a case study.

**Case Study: Browser Wars 2.0** In 2008, the leading internet browser was Microsoft's Internet Explorer (IE). This was thought to be a frustrating and outdated browser as IE was not developed to support the growing features demanded by websites, and users were increasing exposed to better user interfaces ($v_p < t$).[34] Google developed an open-source browser, Chromium, upon which Chrome is built. Additionally, they chose to adopt a code-release strategy, allowing them to benefit from the contributions of external developers, and better understand what features users desired.[35] My model suggests that a code-release strategy can be justified by a specific market environment: where the proprietary software has a low quality (small $v_p$) and the mismatch cost for users is very high (large $t$). When this is the case, code-releasing allows the FOSS to capture a large share of the market (6). Therefore, my model captures closely how Chrome managed to become a market leader in under 3 years.[36]

# 5 Surplus Analysis

In this section, I first study how the design of FOSS licenses influence surplus. I then discuss how effective coordinators are at maximising total surplus. For each coordinator, I focus on the situation where they do not strictly prefer perfectly distinct contributions. This is because when coordinators strictly prefer distinct contributions, there is no effect of $l$ on $\gamma$, as the level of coordination cannot increase beyond 1.

## 5.1 Self-Interested Founder

**Corollary 5.** *When FOSS coordinated by Founders have more permissive licenses:*

- *Consumer surplus generated by the FOSS strictly decreases.*
- *Consumer surplus generated by the proprietary software strictly increases.*
- *Firms' profits strictly decreases. (Corollary 1)*

*See A.5 for proof.*

The first result in Corollary 5 describes consumer surplus of FOSS users, and arises for two reasons. First, the decrease in consumer surplus is directly driven by the fall in the number of FOSS users (Corollary 3). Second, because there are less FOSS users, this

[33]Thomas Greene (2001) reports on Microsoft's stance on open-source as a cancer, and Tom Warren (2020) reports of their retraction on this position.

[34]This is documented by Luke Little (2021); Tom Warren (2018) on Andriod Authority and The Verge respectively.

[35]Releasing Chromium's code corresponds to a permissive licensing strategy, $l \geq \frac{3t-2}{2t}$.

[36]As tracked by Statcounter (2022).

leads to an indirect decrease in the quality of the FOSS.

Since a more permissive license means that more users use the proprietary software, we naturally expect that the consumer surplus contributed by the consumption of the proprietary software increases in $l$. Moreover, Firms lower prices when facing intensified location competition (Lemma 1), lower prices also increases consumer surplus from the proprietary software. This explains the second result in Corollary 5 and highlights how the presence of a niche FOSS generates competition to the benefit of proprietary software users.

## 5.2 Volunteering Altruist

Shifting attention to volunteering Altruists, I show that consumer surplus of FOSS users weakly decreases when the FOSS has a more permissive license. Moreover, users of the proprietary software always benefit when the FOSS license is more permissive. Because Founders and Altruists have similar comparative statics (see Proposition 1 and 2), this analysis is qualitatively identical to the analysis of self-interested Founders.

**Corollary 6.** *When FOSS coordinated by Altruists have more permissive licenses:*

- *Consumer surplus generated by the FOSS weakly decreases.*
- *Consumer surplus generated by the proprietary software strictly increases.*
- *Firms' profits strictly decreases. (Corollary 1)*

*See A.5 for proof.*

Corollary 7 shows that Altruists always generate less total surplus than Founders. For intuition, consider a duopoly with two products that do not benefit from network effects. Further suppose that the first product is located at 0 and Firms sets some price. While the second product is free and located somewhere between $\frac{1}{2}$ and 1. Such an equilibrium generates a fixed level of surplus. Now, allow the second product to benefit from network effects. When the network effect is larger, this improves the surplus generated by the second product, hence improving the total surplus in the market. The higher the network effect, the higher the surplus from the second product, and the larger the total surplus.

Using this intuition and Corollary 4, we can see that Founders generate a higher level of total surplus than Altruists.

**Corollary 7.** *When the FOSS is free, a higher level of coordination improves total surplus.*

*See A.5 for proof.*

Formally, Corollary 7 shows that higher levels of coordination improves total surplus when $p_o = 0$. Since $p_o = 0$ when a FOSS is coordinated by Founders and Altruists,

applying Corollary 4 implies Founders generate more total surplus than Altruists. By extension, Altruists never maximises total surplus.

Together, Corollary 5 and 6 suggest government funding for FOSS — in the name of Digital Sovereignty (See footnote 12.), or to fund competitors to existing proprietary software (e.g. France (Paulina Grzegorzewska, 2021)) — should not judge the success of FOSS based on the number of FOSS users. Instead, even a small and niche FOSS can create sufficient threat, driving down prices of proprietary software; and improve consumer and total surplus. Additionally, Corollary 7 highlights how such funding should perhaps be directly placed in the hands of Founders, instead of through umbrella funding organisations.

## 5.3 Profit-Driven Manager

When studying the surplus generated by profit-driven Managers there is an additional strategic effect stemming from the prices chosen by Managers. Unlike the previous two cases, consumer surplus generated by both the proprietary software and the FOSS are non-monotone. Hence, I only provide an analysis of the profits of Firms and Managers, and the total consumer surplus.

**Corollary 8.** *When FOSS coordinated by Managers have more permissive licenses:*

- *Total consumer surplus strictly decreases.*
- *Profits of Firms weakly decreases, strictly decreasing when* $l \geq 0.5$ *or* $\gamma^M = 1$. *(Corollary 1)*
- *Managers' profits:*
  - *Weakly decreases if* $\gamma^M < 1$*, strictly decreasing when* $l \geq 0.5$.
  - *Strictly increases when* $\gamma^M = 1$.

*See A.5 for proof.*

Corollary 8 points out how, in an environment with costless coordination, Managers may benefit from more permissive FOSS licenses when proprietary software are of sufficiently low quality. Additionally, Corollary 8 also shows how more permissive licenses always harms consumers when Managers coordinate the FOSS.

Indeed, Managers are often in a position to influence what is recognised as an open-source license. The Open Source Initiative (OSI), an organisation which provides an 'independent' stamp of approval on what licenses are deemed 'Open-Source', allows organisations that participates in FOSS development to directly propose members to its governing body. This means Managers are effectively able to select the board of the OSI. In this way, the incentives of the OSI may be potentially skewed in favor of Managers.

Hence, these results question the motives of the OSI, suggesting that tight rules gov-

erning the definition of what is considered 'Open-Source', and the prominence of highly permissive FOSS licenses could be a result of influence from for-profit organisations.

Further scrutinising the motives of the OSI, Corollary 9 shows that Managers do not maximise total surplus.

**Corollary 9.** *When $v_p > t$, Managers do not maximise total surplus.*

*See A.5 for proof.*

# 6 Extensions

I provide a series of extensions, considering the situations where: (i) Founders decide on a license choice instead of level of coordination; (ii) there exists a monopoly of either a FOSS or proprietary software; (iii) developers have heterogeneous skill levels; (iv) users and developers are mutually exclusive groups.

## 6.1 Founder's License Choice

In the main model, I assumed that coordinators are unable to influence the license adopted by a FOSS, taking it as a given. I consider this a realistic setting because many developers are told to adopt 'standard' FOSS licenses such as the MIT license. Unlike coordination effort, once a license has been selected, it is sticky. Hence, as the market environment changes, or as coordination changes hands, licenses are entrenched, and it is more interesting to focus on how the level of coordination evolves. However, it is true that Founders do make an initial license choice at the beginning of a FOSS. To capture the initial license decision, I consider the situation where Founders does not select the level of coordination, instead selecting the license adopted by the FOSS.

**Proposition 4.** *Founders choice of FOSS license is binary. Founders prefer the most permissive license only when the proprietary software has a sufficiently high quality and the marginal quality network effect ($\gamma$) sufficiently large.*[37] *Otherwise, Founders prefer the most restrictive license ($l = 0$).*

*See B.1 for proof.*

Proposition 4 shows that self-interested Founders prefer restrictive licenses under most conditions. They only select permissive licenses when they need to rely on location network effects to compete with a high quality proprietary product.

This result has two interesting features. First, permissive licenses are only selected if quality network effects are sufficiently small, suggesting that location network effects become more important to attract the marginal user. Second, permissive licenses are

[37] Mathematically, $v_p > \frac{t^2(6-4l+l^2)+\gamma^2-2t\gamma(3-l)}{2(t-\gamma)}$ and $\gamma > t$.

only selected when there is a 'good' proprietary software, suggesting that 'new' and breakthrough technological developments will still be largely conducted behind closed doors, and with restrictive licenses.

Importantly, this result suggest that the one size fits all recommendation of the MIT license might be a natural equilibrium. Because FOSS are often developed in competition to other proprietary software, from an initial point, the quality of the proprietary software may be seen as sufficiently high to merit selecting a permissive license. Selecting a permissive license best allows Founders to take advantage of the virtuous cycle of network effects. Hence, despite its critics, it might be easier for Founders to converge upon the MIT license when trying to start a FOSS.

## 6.2 Monopolistic Market

I imposed a restriction of duopoly in the market (Restriction 2). While this provides for a rich analysis, there exists some markets where FOSS, or proprietary software do not exist. Here, I relax Restriction 2, and study what market conditions result in a monopoly being formed.

**Corollary 10.** *Self-interested Founders and Volunteering Altruists never form monopoly. While profit-driven Managers are monopolists when the quality of the proprietary software is low, $v_p \leq t(2l-1)$, and proprietary Firms are monopolist whenever the quality of the proprietary software is high, $v_p \geq t(3-l)$.*

*See B.2 for proof.*

It is interesting to observe that FOSS coordinated by Founders and Altruists are never able to form a monopoly. Such coordinators attempt to minimise some function of mismatch cost, and intrinsically concern themselves with the outcome of the extreme individual, located at 1. Hence, never choose to form a monopoly as this would increase the mismatch cost of the extreme individual.

On the other hand, Managers do not directly concern themselves with mismatch costs. Instead, they seek a balance between demand and setting prices. These are affected by $v_p$ in the same way. Hence, when the proprietary software has a sufficiently low quality, Managers are able to capture a larger share of the market despite raising prices.

## 6.3 Skilled Users

An interesting observation made by Lerner and Tirole (2002) suggests that self-interested Founders tend to be highly skilled, and developers of FOSS tend to be more skilled than the average developer. To reflect this observation, I propose the following modification: Developers are heterogeneous in skill level and this is uniformly distributed according to their location along the Hotelling line. In other words, developers located closer to 1 have

a higher ability, normalised to $\alpha > 0$ at 1, and those further away lower ability, where the user located at 0 has 0 ability. I assume that if a developer chooses to contribute to the FOSS, their ability limits the level of their contribution.

This brings about two effects. First, the value of the software is dependent on the total skill of developers, with developers located closer to 1 generating more value, while those further away generate less value. This means the value of the FOSS is $v_o = \gamma \int_{\alpha\bar{x}}^{\alpha} \frac{x}{\alpha} dx = \gamma \frac{\alpha(1-\bar{x}^2)}{2}$. Second, because developers with a higher skill level are more likely to contribute to more components/critical components of the software, the location of the software is more affected by their output. This means the FOSS is located at $L_o = 1 - l \int_{\alpha\bar{x}}^{\alpha} \frac{x}{\alpha} dx = 1 - \frac{\alpha(1-\bar{x}^2)}{2}$.

**Proposition 5.** *When there exist heterogeneous skill levels, self-interested Founders prefer contributions to be unique, $\gamma^S = 1$.*

*See B.3 for proof.*

Proposition 5 suggests that when there are heterogeneous skill levels, with more skilled developers located further away from the proprietary software, Founders prefers unique contributions. This is because the location of the software shifts smaller for each additional developer than in the base model. As such, the mismatch cost to Founders increases at a lower rate than the base model, and Founders prefer more distinct contributions.

## 6.4 Non-User Developers

I begun with a simplified model where users are also developers. Here, I consider the opposite extreme, where users and developers are mutually exclusive groups. In reality, developers are often a subset of users, and the groups do not perfectly overlap nor are they mutually exclusive. In showing these extremes, I expect that when developers are a subset of users, such a model will sit in-between the results shown in the main model and here.

Recall from Section 3 that users utility is $u_i = v_i - p_i - t|L_i - x|$. $v_i$ captures the quality of the software, in particular $v_o = \gamma \cdot D_D$ where $\gamma$ is the coordinator's effort to manage developers and $D_D$ is the mass of developers.

Also recall that developers exhibit preferences $o \sim U(0, 1)$ along the same Hotelling line as users. Their utility is $w_o = s_o - k|L_o - o|$, $s_o$ captures benefit to the developer, $k > 0$ the cost associated with working on a FOSS with features they do not prefer. As suggested in the literature, developers may be motivated by skill development, and recognition both within the community and outside the community (Hars & Ou, 2002; Mustonen, 2003; Li, Tan, & Teo, 2012). The opportunities to work on complex tasks, or to gain prominence demand on the number of users of a FOSS. Hence, $s_o = \beta \cdot D_o$, where

$\beta$ proxies the prominence and recognition each additional FOSS user brings.

I focus on the situation where the FOSS is coordinated by self-interested Founders. The objective of Founders is given by $\pi_o = \gamma \cdot D_o - t(l \cdot D_o)$, and is identical as the main model.

The timing of the game is identical to the main model, Founders choose the level of coordination, $\gamma$, proprietary Firm sets the price, $p_p$, users and developers simultaneously decide between consuming the proprietary software or the FOSS and contributing to the FOSS respectively.

**Proposition 6.** *Founders only coordinate FOSS in one of the following two situations:*

- *Proprietary software is low quality, $v_p < t$ and the FOSS license is restrictive $l < \frac{1}{2}$;*
- *Proprietary software is high quality, $v_p > t$ and the FOSS license is permissive $l > \frac{1}{2}$,*

*preferring distinct contributions when they do so, $\gamma^D = 1$.*

Proposition 6 highlights how the ability to attract users, and hence developers, can influence when Founders are active. When $v_p < t$, the proprietary software is less competitive, making it easier for the FOSS to attract users. In such circumstances, Founders, maximising their own utility, prefer restrictive licenses as this minimises their mismatch cost. Conversely, when $v_p > t$, and it is more difficult to attract users. Here, Founders need to appeal to a broader audience in order to develop the FOSS. Hence, they are only active if the FOSS license is sufficiently permissive.

# 7 Conclusion

This paper provides a fresh look at the development of Free and Open-Source Software. I account for the different motivations of each actor involved in the development process, using a stylised model with quality and location network effects. In addition to the traditional view of quality network effects, I argue that permissive licenses induce location network effects, and this promotes competition.

I discuss how coordinators with different motivations choose to develop FOSS, finding that self-interested Founders and Volunteering Altruists only coordinate FOSS when licenses are sufficiently restrictive, while profit-driven Managers only coordinate FOSS when licenses are sufficiently permissive.

This suggests that requiring coordinators to adopt permissive licenses (as a result of funding or social norms), can lead to fewer FOSS being developed by Founders and Altruists. Instead, this promotes profit seekers to manage FOSS, cannibalising on community efforts.

Even when active, Founders prefer lower levels of coordination when licenses are more permissive. As a result, permissive licenses can fail to meet the objectives of funders: To develop alternatives to proprietary software or to build a repository of public knowledge. Instead, contributing to the proliferation of niche FOSS which serve very small communities.

However, permissive licenses which lead to the formation of FOSS serving small and niche communities can still drive competition. Because of the dual network effects at play, proprietary Firms prefer to significantly reduce the price of proprietary software to limit both the quality and location competitive effects from FOSS. This suggests that in an environment with FOSS, market concentration can be a poor judgement of market competitiveness.

I show that Founders generate more surplus than Altruists. Hence, purely from a motivation standpoint, directing funding to larger FOSS communities or umbrella organisations can result in a less efficient user of resources. Instead, Founders are better suited to develop any particular FOSS. This puts into question how funding is secured and directed through umbrella organisations, such as the Apache and Linux foundations, and supports the idea of direct funding for individual FOSS projects.

At its heart, this paper is about the motivations behind social movements, organised efforts by a group of people to achieve a common goal. Social movements often originate from extremely dissatisfied persons who find it beneficial to coordinate a group of like-minded individuals to address a shared dissatisfaction (Breton & Breton, 1969). As groups grow, they become more valuable to their members. However, differences of opinions are more likely to manifest with larger groups. This friction can diminish the value of the group to an individual. Therefore, in addition to Free and Open-Source Software, location and quality network effects co-exist in other social movements such as crowd-sourcing and resource sharing, online gaming communities, social and political activism. I leave such applications for future research.

# References

Adem Ait Fonollà. (2022). *Survival rate of github projects – an empirical study.* Retrieved 2023-07, from https://livablesoftware.com/survival-rate-github-projects-empirical/

Almeida, D. A., Murphy, G. C., Wilson, G., & Hoye, M. (2017). Do software developers understand open source licenses? In *2017 ieee/acm 25th international conference on program comprehension (icpc)* (pp. 1–11).

Andersen-Gott, M., Ghinea, G., & Bygstad, B. (2012). Why do commercial companies contribute to open source software? *International journal of information management*, *32*(2), 106–117.

André Staltz. (2019). *Software below the poverty line.* Retrieved 2023-05, from https://staltz.com/software-below-the-poverty-line.html

Athey, S., & Ellison, G. (2014). Dynamics of open source movements. *Journal of Economics & Management Strategy*, *23*(2), 294–316.

August, T., Chen, W., & Zhu, K. (2021). Competition among proprietary and open-source software firms: The role of licensing in strategic contribution. *Management Science*, *67*(5), 3041–3066.

Balter, B. (2015). *Open source license usage on github.com.* Retrieved 2023-03, from https://github.blog/2015-03-09-open-source-license-usage-on-github-com/

Belleflamme, P., & Peitz, M. (2015). *Industrial organization: markets and strategies.* Cambridge University Press.

Benjamin Cedric Larsen. (2022). *The geopolitics of ai and the rise of digital sovereignty.* Retrieved 2023-03, from https://www.brookings.edu/articles/the-geopolitics-of-ai-and-the-rise-of-digital-sovereignty/

Besley, T. (1995). Property rights and investment incentives: Theory and evidence from ghana. *Journal of political Economy*, *103*(5), 903–937.

Blind, K., & Schubert, T. (2023). Estimating the gdp effect of open source software and its complementarities with r&d and patents: evidence and policy implications. *The Journal of Technology Transfer*, 1–26.

Bloxam, G. (1957). Letters patent for inventions: Their use and misuse. *The Journal of Industrial Economics*, *5*(3), 157–179.

Boone, J. (2001). Intensity of competition and the incentive to innovate. *International Journal of Industrial Organization*, *19*(5), 705–726.

Boston Consulting Group. (2021). *Why you need an open source software strategy.* Retrieved 2023-05, from https://www.bcg.com/publications/2021/open-source-software-strategy-benefits

Breton, A., & Breton, R. (1969). An economic theory of social movements. *The American*

*Economic Review*, *59*(2), 198–205.

Brewster, R. (2009). *A new license for paint.net v3.5.* Retrieved 2023-03, from https://blog.getpaint.net/2009/11/06/a-new-license-for-paintnet-v35/

Brian Proffitt. (2016). *Hierarchy in open source.* Retrieved 2023-03, from https://www.redhat.com/en/blog/hierarchy-open-source

Casadesus-Masanell, R., & Ghemawat, P. (2006). Dynamic mixed duopoly: A model motivated by linux vs. windows. *Management Science*, *52*(7), 1072–1084.

Chen, Y., & Puttitanun, T. (2005). Intellectual property rights and innovation in developing countries. *Journal of development economics*, *78*(2), 474–493.

Chris Stokel-Walker. (2014). *The internet is being protected by two guys named steve.* Retrieved 2023-05, from https://www.buzzfeed.com/chrisstokelwalker/the-internet-is-being-protected-by-two-guys-named-st

Chris Williams. (2016). *How one developer just broke node, babel and thousands of projects in 11 lines of javascript.* Retrieved 2023-05, from https://www.theregister.com/2016/03/23/npm_left_pad_chaos/

d'Aspremont, C., Gabszewicz, J. J., & Thisse, J.-F. (1979). On hotelling's "stability in competition". *Econometrica*, *47*(5), 1145–1150.

Denis Pushkarev. (2023). *2023-02-14-so-whats-next.md.* Retrieved 2023-05, from https://github.com/zloirock/core-js/blob/master/docs/2023-02-14-so-whats-next.md

Emma Roth. (2022). Open source developer corrupts widely-used libraries, affecting tons of projects. *The Verge*. Retrieved 2022-10, from https://www.theverge.com/2022/1/9/22874949/developer-corrupts-open-source-libraries-projects-affected

Etzion, H., & Pang, M.-S. (2014). Complementary online services in competitive markets: Maintaining profitability in the presence of network effects. *Mis Quarterly*, *38*(1), 231–248.

Fainmesser, I. P., & Galeotti, A. (2020). Pricing network effects: Competition. *American Economic Journal: Microeconomics*, *12*(3), 1–32.

Fielding, R. T. (1999). Shared leadership in the apache project. *Communications of the ACM*, *42*(4), 42–43.

Franke, N., & Von Hippel, E. (2003). Satisfying heterogeneous user needs via innovation toolkits: the case of apache security software. *Research policy*, *32*(7), 1199–1215.

Gamblin, T. (2021). *Picking an open source license at llnl: Guidance and recommendations from the computing directorate* (Tech. Rep.). Livermore, CA (United States): Lawrence Livermore National Lab.(LLNL).

Gaudeul, A. (2005). Public provision of a private good: What is the point of the bsd license? *Available at SSRN 933631*.

Gaudeul, A. (2008). Open source licensing in mixed markets, or why open source software

does not succeed. *CCP Working Paper*.

Hars, A., & Ou, S. (2002). Working for free? motivations for participating in open-source projects. *International journal of electronic commerce*, *6*(3), 25–39.

Hotelling, H. (1929). Stability in competition. *The Economic Journal*, *39*(153), 41–57.

Janis Lesinski. (2020). *The mit licence is community hostile.* Retrieved 2023-05, from `https://www.lesinskis.com/MIT-licence-community-hostile.html`

John Biggs. (2014). *Heartbleed, the first security bug with a cool logo.* Retrieved 2023-05, from `https://techcrunch.com/2014/04/09/heartbleed-the-first-consumer-grade-exploit/`

Johnson, J. P. (2002). Open source software: Private provision of a public good. *Journal of Economics & Management Strategy*, *11*(4), 637–662.

Johnson, J. P. (2006). Collaboration, peer review and open source software. *Information Economics and Policy*, *18*(4), 477–497.

Klug, D., Bogart, C., & Herbsleb, J. D. (2021). "they can only ever guide" how an open source software community uses roadmaps to coordinate effort. *Proceedings of the ACM on Human-Computer Interaction*, *5*(CSCW1), 1–28.

Lakhani, K. R., & Hippel, E. v. (2004). How open source software works:"free" user-to-user assistance. In *Produktentwicklung mit virtuellen communities* (pp. 303–339). Springer.

Lerner, J., & Tirole, J. (2002). Some simple economics of open source. *The journal of industrial economics*, *50*(2), 197–234.

Lerner, J., & Tirole, J. (2005a). The economics of technology sharing: Open source and beyond. *Journal of Economic Perspectives*, *19*(2), 99–120.

Lerner, J., & Tirole, J. (2005b). The scope of open source licensing. *Journal of Law, Economics, and Organization*, *21*(1), 20–56.

Li, Y., Tan, C.-H., & Teo, H.-H. (2012). Leadership characteristics and developers' motivation in open source software development. *Information & Management*, *49*(5), 257–267.

Luke Little. (2021). *Google chrome: Its history and rise to market domination.* Retrieved 2022-11, from `https://www.androidauthority.com/google-chrome-history-1025602/`

Madiega, T. A. (2020). Digital sovereignty for europe. *EPRS: European Parliamentary Research Service*.

Michlmayr, M., Hunt, F., & Probert, D. (2007). Release management in free software projects: Practices and problems. In *Open source development, adoption and innovation: Ifip working group 2.13 on open source software, june 11–14, 2007, limerick, ireland 3* (pp. 295–300).

Microsoft. (2023a). *Microsoft 365 roadmap.* Retrieved 2023-05, from `https://www.microsoft.com/en-gb/microsoft-365/roadmap?filters=`

Microsoft. (2023b). *Microsoft release cycle.* Retrieved 2023-05, from https://learn.microsoft.com/en-us/windows/deployment/update/release-cycle

Moser, P. (2005). How do patent laws influence innovation? evidence from nineteenth-century world's fairs. *American economic review*, *95*(4), 1214–1236.

Mustonen, M. (2003). Copyleft—the economics of linux and other open source software. *Information Economics and Policy*, *15*(1), 99–121.

Nicolas Suzor. (2013). *What motivates free software developers to choose between copyleft and permissive licences?* Retrieved 2023-05, from https://opensource.com/law/13/8/motivation-free-software-licensing

Open Source Initiative. (2022). *Licenses & standards.* Retrieved 2022-11, from https://opensource.org/licenses

Paulina Grzegorzewska. (2021). *Parlimentary mission supports open source.* Retrieved 2023-03, from https://joinup.ec.europa.eu/collection/open-source-observatory-osor/news/french-report-digital-sovereignty

Qian, Y. (2007). Do national patent laws stimulate domestic innovation in a global patenting environment? a cross-country analysis of pharmaceutical patent protection, 1978–2002. *The Review of Economics and Statistics*, *89*(3), 436–453.

Raymond, E. (1999). The cathedral and the bazaar. *Knowledge, Technology & Policy*, *12*(3), 23–49.

Robles, G., Steinmacher, I., Adams, P., & Treude, C. (2019). Twenty years of open source software: From skepticism to mainstream. *IEEE Software*, *36*(6), 12–15.

Sean Fleming. (2021). *What is digital sovereignty and why is europe so interested in it?* Retrieved 2023-03, from https://www.weforum.org/agenda/2021/03/europe-digital-sovereignty/

Shah, S. K. (2006). Motivation, governance, and the viability of hybrid forms in open source software development. *Management science*, *52*(7), 1000–1014.

Shy, O., & Thisse, J.-F. (1999). A strategic approach to software protection. *Journal of Economics & Management Strategy*, *8*(2), 163–190.

Stackexchange. (2015). *How can a project be relicensed?* Retrieved 2023-05, from https://opensource.stackexchange.com/questions/33/how-can-a-project-be-relicensed

Statcounter. (2022). *Desktop browser market share worldwide.* Retrieved 2022-11, from https://gs.statcounter.com/browser-market-share/desktop/worldwide#monthly-200901-202210

Subramaniam, C., Sen, R., & Nelson, M. L. (2009). Determinants of open source software project success: A longitudinal study. *Decision Support Systems*, *46*(2), 576–585.

The Apache Software Foundation. (2023a). *Apache corporate governance.* Retrieved 2023-03, from https://www.apache.org/foundation/governance/

The Apache Software Foundation. (2023b). *Foundation reports and statements.* Retrieved

2023-05, from https://www.apache.org/foundation/reports.html

The Document Foundation. (2023). *The document foundation.* Retrieved 2023-03, from https://www.documentfoundation.org/governance/bodies/

The Linux Foundation. (2023). *Linux foundation leadership teams.* Retrieved 2023-03, from https://www.linuxfoundation.org/about/leadership

Thomas Greene. (2001). *Ballmer: 'linux is a cancer'.* Retrieved 2023-03, from https://www.theregister.com/2001/06/02/ballmer_linux_is_a_cancer/

Tom Warren. (2018). *Google's chrome browser is now 10 years old.* Retrieved 2022-11, from https://www.theverge.com/2018/9/2/17811844/google-chrome-browser-10-years-history

Tom Warren. (2020). *Microsoft: we were wrong about open source.* Retrieved 2023-03, from https://www.theverge.com/2020/5/18/21262103/microsoft-open-source-linux-history-wrong-statement

Vendome, C., Linares-Vásquez, M., Bavota, G., Di Penta, M., German, D., & Poshyvanyk, D. (2015a). License usage and changes: a large-scale study of java projects on github. In *2015 ieee 23rd international conference on program comprehension* (pp. 218–228).

Vendome, C., Linares-Vásquez, M., Bavota, G., Di Penta, M., German, D. M., & Poshyvanyk, D. (2015b). When and why developers adopt and change software licenses. In *2015 ieee international conference on software maintenance and evolution (icsme)* (pp. 31–40).

# Appendix A Proofs

## A.1 Proprietary Firm

To show Lemma 1, I first show the following Lemma:

**Lemma 2.** *The indifferent user is given by* $\bar{x} = \frac{v_p - p_p + p_o - \gamma + t(1-l)}{t(2-l)-\gamma}$. $\frac{\partial \bar{x}}{\partial \gamma} < 0$ *and* $\frac{\partial \bar{x}}{\partial l} < 0$.

*Proof of Lemma 2.*
By Restriction 2, $\bar{x} \in (0, 1)$ and an interior profit-maximising solution implies

$$t(2-l) - \gamma > 0 \tag{a.1}$$

$$v_P - p_p + p_o - \gamma + t(1-l) > 0 \tag{a.2}$$

$$p_p + t - v_p - p_o > 0 \tag{a.3}$$

Therefore,

$$\frac{\partial \bar{x}}{\partial \gamma} = \frac{v_p - t + p_o - p_p}{(t(2-l)-\gamma)^2}$$
$$\frac{\partial \bar{x}}{\partial l} = \frac{t(v_p - t + p_o - p_p)}{(t(2-l)-\gamma)^2}$$

applying (a.3), both $\frac{\partial \bar{x}}{\partial \gamma}$ and $\frac{\partial \bar{x}}{\partial l}$ are $< 0$.

This concludes the proof. □

I now show Lemma 1.

*Proof of Lemma 1.*
To find the equilibrium price of the proprietary software, consider the profit function of Firms.

$$\begin{aligned}\pi_p &= p_p \cdot \bar{x} \\ &= p_p \frac{v_p - p_p + p_o - \gamma + t(1-l)}{t(2-l)-\gamma}\end{aligned}$$

Notice that for an interior profit maximising solution, $t(2-l) - \gamma > 0$ is required. This ensures that $\frac{\partial^2 \pi_p}{\partial p_p^2} < 0$, and implies that $v_p - p_p + p_o - \gamma + t(1-l) > 0$.

Therefore, $p_p^* = \frac{v_p+p_o-\gamma+t(1-l)}{2}$ when Restriction 2 holds. Restriction 2 implies that

$$t(2-l)-\gamma>0$$
$$v_p+p_o+t(1-l)-\gamma>0 \tag{a.4}$$
$$t(3-l)-p_o-v_p-\gamma>0 \tag{a.5}$$

I now show that $p_p^*$ increases in $v_p$ and $p_o$, and decreases in $\gamma$ and $l$.

$$\frac{\partial p_p^*}{\partial v_p}=\frac{1}{2}>0$$
$$\frac{\partial p_p^*}{\partial p_o}=\frac{1}{2}>0$$
$$\frac{\partial p_p^*}{\partial \gamma}=-\frac{1}{2}<0$$
$$\frac{\partial p_p^*}{\partial l}=-\frac{t}{2}<0$$

This concludes the proof. □

*Proof of Corollary 1.*
To show Corollary 1, recall that $\pi_p = p_p \cdot \bar{x}$. Considering (1) and $p_p^*$,

$$\pi_p=\frac{[v_p+p_o-\gamma+t(1-l)]^2}{4(t(2-l)-\gamma)}$$
$$\frac{\partial \pi_p}{\partial \gamma}=\frac{v_p+p_o+t(1-l)-\gamma}{4(t(2-l)-\gamma)^2}(v_p+p_o+\gamma-t(3-l))$$
$$\frac{\partial \pi_p}{\partial l}=t\frac{v_p+p_o+t(1-l)-\gamma}{4(t(2-l)-\gamma)^2}(v_p+p_o+\gamma-t(3-l))=t\frac{\partial \pi_p}{\partial \gamma}$$

First notice that $t(2-l)-\gamma>0$ (a.1) and $v_p+p_o+t(1-l)-\gamma>0$ (a.2). This means that the sign of $\frac{\partial \pi_p}{\partial \gamma}$ depends on the sign of $v_p+p_o+\gamma-t(3-l)$. From (a.5), $v_p+p_o+\gamma-t(3-l)<0$. Therefore $\frac{\partial \pi_p}{\partial \gamma}<0$, and by extension $\frac{\partial \pi_p}{\partial l}<0$.

This concludes the proof. □

## A.2 Self-Interested Founder

I solve the game by backward induction. Notice that in Section A.1, I have solved the last two stages of the game. And this is common to all coordinators. Here, I provide the proof for what happens in the first stage of the game when the coordinator is a Founder.

*Proof of Proposition 1.*
To proof Proposition 1, I first propose a candidate equilibrium, ruling out all other

possible equilibrium. I then show that such an equilibrium satisfies the Restrictions, and maximises Founders' utility.

Recall that $\pi_o = (\gamma - tl)(1 - \bar{x})$. Considering (1), and remembering that $p_o = 0$,

$$\pi_o = \frac{(\gamma - tl)(t(3-l) - v_p - \gamma)}{2(t(2-l) - \gamma)}$$

Taking the first and second order conditions,

$$\begin{aligned}
\frac{\partial \pi_o}{\partial \gamma} &= \frac{t(t(6 - 6l + l^2) - 2v_p(1-l)) + \gamma^2 - 2\gamma t(2-l)}{2(t(2-l) - \gamma)^2} \\
&\quad \text{evaluated at } \gamma = 0, \\
&= \frac{t(6 - 6l + l^2) - 2v_p(1-l)}{2t(2-l)^2}, > 0 \text{ at } t(3 + \frac{l^2}{2(1-l)}) > v_p \\
\frac{\partial^2 \pi_o}{\partial \gamma^2} &= -\frac{2t(1-l)(v_p - t)}{(t(2-l) - \gamma)^3}, < 0 \text{ at } t(2-l) - \gamma > 0.
\end{aligned}$$

Notice that $\gamma \in [0, 1]$ implies the FOSS is only active if $3t + \frac{tl^2}{2(1-l)} > v_p$.

Solving for $\gamma^*$

$$\gamma^* = t(2-l) \pm \sqrt{2t(v_p - t)(1-l)}$$

Notice that $\sqrt{2t(v_p - t)(1-l)}$ is real only if $v_p \geq t$, because $l \in (0, 1)$. And when $\sqrt{2t(v_p - t)(1-l)}$ is real, it is $\geq 0$. Because $t(2-l) > \gamma$ (a.1), I reject $t(2-l) + \sqrt{2t(v_p - t)(1-l)}$ as a candidate for $\gamma^*$. Moreover, because $\gamma \in [0, 1]$, $\gamma^* = 1$ when $t > \frac{1}{2-l}$ and $v_p \leq \frac{1 - 2t(2-l) + t^2(6 - 6l + l^2)}{2t(1-l)} = 3t + \frac{tl^2}{2(1-l)} + \frac{1}{2t(1-l)} - \frac{2-l}{1-l}$. In all other cases, $\gamma^* = t(2-l) - \sqrt{2t(v_p - t)(1-l)}$. Therefore, I propose $\gamma^* = min\{t(2-l) - \sqrt{2t(v_p - t)(1-l)}, 1\}$ as a candidate solution.

I show when this solution satisfies Restriction 1, Consider that Founders and the indifferent user must have weakly positive utility. Founders' utility is given by $\pi_o$, and this is weakly positive whenever $\gamma \geq tl$. To see that $\gamma^* \geq tl$,

$$\begin{aligned}
t(2-l) - \sqrt{2t(v_p - t)(1-l)} &\geq tl \\
2t(1-l) &\geq \sqrt{2t(v_p - t)(1-l)} \\
2t(1-l) &\geq (v_p - t) \\
t(3 - 2l) &\geq v_p \\
\frac{3t - v_p}{2t} &\geq l
\end{aligned}$$

and $1 \geq tl \iff l \leq \frac{1}{t}$ when $\gamma^* = 1$.

The indifferent user's utility is $\frac{(\gamma-t(1-l))(t(3-l)-v_p-\gamma)}{2(t(2-l)-\gamma)}$. Because $t(2-l)-\gamma>0$ (a.1) and $t(3-l)-v_p-\gamma>0$ (a.5), it remains that the indifferent user's utility is weakly positive if $\gamma \geq t(1-l)$. To see that $\gamma^* \geq t(1-l)$,

$$\begin{aligned} t(2-l)-\sqrt{2t(v_p-t)(1-l)} &\geq t(1-l) \\ t &\geq \sqrt{2t(v_p-t)(1-l)} \\ t &\geq 2(v_p-t)(1-l) \\ l &\geq \frac{2v_p-3t}{2(v_p-t)} \end{aligned}$$

and $1 \geq t(1-l) \iff l \geq 1-\frac{1}{t}$ when $\gamma^*=1$.

Combining $\frac{3t-v_p}{2t} \geq l$ with $l \geq \frac{2v_p-3t}{2(v_p-t)}$,

$$\begin{aligned} \frac{3t-v_p}{2t} &\geq \frac{2v_p-3t}{2(v_p-t)} \\ \frac{3t-v_p}{t} &\geq \frac{2v_p-3t}{v_p-t} \\ (3t-v_p)(v_p-t) &\geq t(2v_p-3t) \\ v_p(2t-v_p) &\geq 0 \end{aligned}$$

implies that $2t \geq v_p$.

Together, this means that when $\gamma^*=t(2-l)-\sqrt{2t(v_p-t)(1-l)}$, $l \in [\frac{2v_p-3t}{2(v_p-t)}, \frac{3t-v_p}{2t}]$ such that $2t \geq v_p$. And $1-\frac{1}{t} \leq l \leq \frac{1}{t}$, which is true when $t[1,2]$.

Next, I show when this solution satisfies Restriction 2. It is immediate that the interior candidate solution always satisfies (a.1). To see when it satisfies (a.4),

$$\begin{aligned} v_p+t(1-l)-t(2-l)+\sqrt{2t(v_p-t)(1-l)} &> 0 \\ v_p-t+\sqrt{2t(v_p-t)(1-l)} &> 0 \end{aligned}$$

This is always true.

To see when it satisfies (a.5),

$$\begin{aligned} t(3-l)-v_p-t(2-l)+\sqrt{2t(v_p-t)(1-l)} &> 0 \\ t-v_p+\sqrt{2t(v_p-t)(1-l)} &> 0 \\ 2t(1-l) &> v_p-t \\ t(3-2l) &> v_p \end{aligned}$$

And the corner solution of $\gamma^* = 1$ satisfies these equations when $t(2-l) > 1$, $v_p + t(1-l) > 1$, and $t(3-l) - v_p > 1$.

Given these Restrictions, I now show that the candidate solution is utility maxinimising. For the candidate solution to be utility maximising, it must be that $\frac{\partial \pi_o}{\partial \gamma} > 0$ at $\gamma = 0$ and $\frac{\partial^2 \pi_o}{\partial \gamma^2} < 0$.

At $\gamma = 0$, $\frac{\partial \pi_o}{\partial \gamma}$ is $\frac{t(6-6l+l^2)-2v_p(1-l)}{2t(2-l)^2}$. This is positive whenever $t(3 + \frac{l^2}{2(1-l)}) > v_p$. Recall from (a.5) that $t + \sqrt{2t(v_p - t)(1-l)} > v_p$. I now show directly that $t(3 + \frac{l^2}{2(1-l)}) \geq t + \sqrt{2t(v_p - t)(1-l)}$:

$$t(3 + \frac{l^2}{2(1-l)}) \geq t + \sqrt{2t(v_p - t)(1-l)}$$
$$t(2 + \frac{l^2}{2(1-l)}) \geq \sqrt{2t(v_p - t)(1-l)}$$

Recall that $t(3-2l) > v_p \iff 2t(1-l) > v_p - t$. This means that $2t(1-l) > \sqrt{2t(v_p - t)(1-l)}$ and showing $t(2 + \frac{l^2}{2(1-l)}) \geq 2t(1-l)$ implies that $t(2 + \frac{l^2}{2(1-l)}) > \sqrt{2t(v_p - t)(1-l)}$.

$$t(2 + \frac{l^2}{2(1-l)}) \geq 2t(1-l)$$
$$\frac{4(1-l) + l^2}{2(1-l)} \geq 2(1-l)$$
$$4 \geq 3l$$

Since $l \in (0, 1)$, this is always true.

It is immediate that $\frac{\partial^2 \pi_o}{\partial \gamma^2} < 0$ for the range of possible solutions. This is because it is negative whenever $v_p > t$ is required for a real solution, and (a.1) must be satisfied, $l \in (0, 1)$, and $t > 0$.

Therefore, whenever Restriction 2 is satisfied, the candidate solution maximises the utility of Founders.

I now show that Founders do not always play $\gamma = 1$ when it is active. For Founders to always play $\gamma^* = 1$, it must be that $3t + \frac{tl^2}{2(1-l)} + \frac{1}{2t(1-l)} - \frac{2-l}{1-l} \geq 3t + \frac{tl^2}{2(1-l)}$, and $t > \frac{1}{2-l}$. Suppose to a contradiction that Founders always plays $\gamma^* = 1$.

The first criteria reduces to

$$\begin{aligned} 3t + \frac{tl^2}{2(1-l)} + \frac{1}{2t(1-l)} - \frac{2-l}{1-l} &\geq 3t + \frac{tl^2}{2(1-l)} \\ \frac{1}{2t(1-l)} - \frac{2-l}{1-l} &\geq 0 \\ \frac{1-2t(2-l)}{t(1-l)} &\geq 0 \\ \frac{1}{2(2-l)} &\geq t \\ \frac{1}{2(2-l)} &> \frac{1}{2-l} \\ \frac{1}{2} &> 1 \end{aligned}$$

A contradiction. This means that there exists a range of $v_p$ and $t$ for which Founders will play the $t(2-l) - \sqrt{2t(v_p-t)(1-l)}$.

Therefore, I conclude that there exists a unique equilibrium where $\gamma^* = min\{t(2-l) - \sqrt{2t(v_p-t)(1-l)}, 1\}$ maximises Founders' utility subject to Restriction 1 and 2, which require $l \in [\frac{2v_p-3t}{2(v_p-t)}, \frac{3t-v_p}{2t}]$. In particular, $\gamma^* = 1$ if $v_p \leq 3t + \frac{tl^2}{2(1-l)} + \frac{1}{2t(1-l)} - \frac{2-l}{1-l}$ and $t > \frac{1}{2-l}$, and $l \in [1-\frac{1}{t}, \frac{1}{t}]$.

To see that $\frac{\partial \gamma^*}{\partial l} \leq 0$ and $\frac{\partial \gamma^*}{\partial v_p} \leq 0$, at $\gamma^* = t(2-l) - \sqrt{2t(v_p-t)(1-l)}$,

$$\begin{aligned} \frac{\partial \gamma^*}{\partial l} &= \frac{t(v_p - t - \sqrt{2t(v_p-t)(1-l)})}{\sqrt{2t(v_p-t)(1-l)}} \\ \frac{\partial \gamma^*}{\partial v_p} &= -\frac{t(1-l)}{\sqrt{2t(v_p-t)(1-l)}} \end{aligned}$$

From (a.5), $t - v_p + \sqrt{2t(v_p-t)(1-l)} > 0$, hence $\frac{\partial \gamma^*}{\partial l} < 0$. And $l \in (0,1)$ implies $\frac{\partial \gamma^*}{\partial v_p} < 0$.

When $\gamma^* = 1$, changes to $l$ or $v_p$ does not affect $\gamma^*$. Therefore $\frac{\partial \gamma^*}{\partial l} = 0$ and $\frac{\partial \gamma^*}{\partial v_p} = 0$.

This concludes the proof. □

*Proof of Corollary 2.*
To show Corollary 2, recall that

$$\pi_o = \frac{(\gamma - tl)(t(3-l) - v_p - \gamma)}{2(t(2-l) - \gamma)}$$

Notice that Founders only benefit if $\gamma > tl$ and $t(3-l) - v_p - \gamma > 0$. This means that Founders only manage the FOSS if $v_p < t(3-2l)$, and the most permissive license that an active FOSS can use is $l < \frac{3t-v_p}{2t} = \bar{l}$. If a license is too permissive, Firms form a

monopolist in the market.

$$\frac{\partial \bar{l}}{\partial v_p} = -\frac{1}{2t} < 0$$
$$\frac{\partial \bar{l}}{\partial t} = \frac{3t + v_p}{2t} > 0$$

This concludes the proof. □

## A.3 Volunteering Altruist

Recall that in Section A.1 that I have solved the last two stages of the game, hence I rely on these results to proof the first stage of the game.

*Proof of Proposition 2.*
To proof Proposition 2, I first propose a candidate equilibrium, ruling out all other possible equlibria. I then show that the candidate equilibrium satisfies the Restrictions, and maximises Altruists' utility.

Recall that $\pi_o^A = \int_{\bar{x}}^1 \gamma(1-\bar{x}) - t|1 - l(1-\bar{x}) - x|dx$. Accounting for (1), $\pi_o^A = \frac{(2\gamma - t(1-2l+2l^2))(v_p+\gamma-t(3-l))^2}{8(t(2-l)-\gamma)^2}$.

Taking the first order condition,

$$\frac{\partial \pi_o^A}{\partial \gamma} = \frac{(v_p + \gamma - t(3-l))(\gamma(v_p + t(3-2l) - \gamma) + t(v_p(1+l-2l^2) - t(5-3l-l^2)))}{4(t(2-l)-\gamma)^3}$$
$$\frac{\partial^2 \pi_o^A}{\partial \gamma^2} = \frac{(v_p - t)(4t\gamma(1-l^2) - t^2(17-4l-14l^2+4l^3) + 2\gamma v_p + t v_p(5+2l-6l^2))}{4(t(2-l)-\gamma)^4}$$

First, I find the candidate $\gamma^A$. Solving $\frac{\partial \pi_o^A}{\partial \gamma} = 0$,

$$\gamma^A = \begin{cases} t(3-l) - v_p \\ \frac{t(3-2l)+v_p - \sqrt{(v_p - t)(11t - 8tl^2 + v_p)}}{2} \\ \frac{t(3-2l)+v_p + \sqrt{(v_p - t)(11t - 8tl^2 + v_p)}}{2} \end{cases}$$

I eliminate the first candidate because it violates $t(3-l) - v_p - \gamma > 0$ (a.5). I eliminate

the third candidate because $v_p + t(1-l) - \gamma > 0$,

$$\begin{aligned} v_p + t(1-l) - \frac{t(3-2l) + v_p + \sqrt{(v_p - t)(11t - 8tl^2 + v_p}}{2} &> 0 \\ \frac{v_p - t - \sqrt{(v_p - t)(11t - 8tl^2 + v_p)}}{2} &> 0 \\ (v_p - t)^2 &> (v_p - t)(11t - 8tl^2 + v_p) \\ 8tl^2 &> 12t \\ 2l^2 &> 3 \end{aligned}$$

This is not possible because $l \in (0,1)$. This leaves $\frac{t(3-2l)+v_p-\sqrt{(v_p-t)(11t-8tl^2+v_p)}}{2}$ as the only candidate. Since $(11t - 8tl^2 + v_p) > 0$, a real solution only exists if $v_p > t$.

I now show that at the candidate $\gamma^A = \frac{t(3-2l)+v_p-\sqrt{(v_p-t)(11t-8tl^2+v_p)}}{2}$ that $\frac{\partial \pi_o^A}{\partial \gamma} < 0$. Notice that $\frac{\partial \pi_o^A}{\partial \gamma} > 0$ only if $v_p < t\frac{5-2l^2}{2}$ and $\gamma > \frac{t(t(17-4l-14l^2+4l^3)-v_p(5+2l-6l^2))}{2(v_p+2(1-l^2)t)}$, and $\frac{\partial \pi_o^A}{\partial \gamma} < 0$ whenever $v_p > t\frac{5-2l^2}{2}$ or $\gamma < \frac{t(t(17-4l-14l^2+4l^3)-v_p(5+2l-6l^2))}{2(v_p+2(1-l^2)t)}$. I show that the candidate $\gamma^A < \frac{t(t(17-4l-14l^2+4l^3)-v_p(5+2l-6l^2))}{2(v_p+2(1-l^2)t)}$ when $v_p < t\frac{5-2l^2}{2}$. Therefore there is no scenario where $\frac{\partial^2 \pi_o^A}{\partial \gamma^2} > 0$.

$$\begin{aligned} \frac{t(3-2l) + v_p - \sqrt{(v_p - t)(11t - 8tl^2 + v_p)}}{2} &< \frac{t(t(17 - 4l - 14l^2 + 4l^3) - v_p(5 + 2l - 6l^2))}{2(v_p + 2(1 - l^2)t)} \\ \frac{(v_p - t)(11t - 8tl^2 + v_p)}{v_p + 2t(1 - l^2)} &< \sqrt{(v_p - t)(11t - 8tl^2 + v_p)} \\ \frac{(v_p - t)(11t - 8tl^2 + v_p)}{(v_p + 2t(1 - l^2))^2} &< 1 \\ (v_p - t)(11t - 8tl^2 + v_p) &< (v_p + 2t(1 - l^2))^2 \\ t(3 - 2l^2)(2v_p + t(-5 + 2l^2)) &< 0 \text{ notice that } 3 - 2l^2 > 0 \\ 2v_p - t(5 - 2l^2) &< 0 \end{aligned}$$

Therefore, whenever $v_p < t\frac{5-2l^2}{2}$, $\gamma^A < \frac{t(t(17-4l-14l^2+4l^3)-v_p(5+2l-6l^2))}{2(v_p+2(1-l^2)t)}$. And I conclude that there is no scenario where $\frac{\partial^2 \pi_o^A}{\partial \gamma^2} > 0$ subject to my Restrictions. And the candidate equilibrium maximises Altruists' utility.

However, notice that there exists some range where $\gamma^A = 1$. This upper bound is reached when $\frac{t(3-2l)+v_p-\sqrt{(v_p-t)(11t-8tl^2+v_p)}}{2} \geq 1 \iff v_p \leq \frac{1+t^2(5-3l-l^2)-t(3-2l)}{1+t(1+l-2l^2)}$ and $t > \frac{1}{2-l}$.

I conclude that Altruists always selects a positive level of coordination, $\gamma^A > 0$. And $\gamma^A = min\{\frac{t(3-2l)+v_p-\sqrt{(v_p-t)(11t-8tl^2+v_p)}}{2}, 1\}$ and Altruists are active when $v_p < t\frac{5-2l^2}{2}$.

To see that $\frac{\partial \gamma^A}{\partial v_p} \leq 0$, at $\frac{t(3-2l)+v_p-\sqrt{(v_p-t)(11t-8tl^2+v_p)}}{2}$,

$$\frac{\partial \gamma^A}{\partial v_p} = \frac{1}{2} - \frac{v_p + t(5-4l^2)}{2\sqrt{(v_p-t)(11t-8tl^2+v_p)}}$$

To see that

$$\begin{aligned}
\frac{1}{2} - \frac{v_p + t(5-4l^2)}{2\sqrt{(v_p-t)(11t-8tl^2+v_p)}} &< 0 \\
\sqrt{(v_p-t)(11t-8tl^2+v_p)} &< v_p + t(5-4l^2) \\
(v_p-t)(11t-8tl^2+v_p) &< (v_p + t(5-4l^2))^2 \\
0 &< t^2(3-2l^2)^2
\end{aligned}$$

this is always true because $t > 0$ and $l \in (0, 1)$.

And when $\gamma^A = 1$, changes to $v_p$ does not affect $\gamma^A$. Therefore, $\frac{\partial \gamma^A}{\partial v_p} \leq 0$.

This concludes the proof. □

*Proof of Corollary 4.*
To see this, I compare $\gamma^*$ with $\gamma^A$ whenever $\gamma^* < 1$ and $\gamma^A < 1$.

$$\begin{aligned}
\gamma^* &> \gamma^A \\
t(2-l) - \sqrt{2t(v_p-t)(1-l)} &> \frac{t(3-2l) + v_p - \sqrt{(v_p-t)(v_p+11t-8tl^2)}}{2} \\
t - 2\sqrt{2t(v_p-t)(1-l)} &> v_p - \sqrt{(v_p-t)(v_p+11t-8tl^2)} \\
t - v_p + \sqrt{(v_p-t)(v_p+11t-8tl^2)} &> 2\sqrt{2t(v_p-t)(1-l)}
\end{aligned}$$

Notice that $\sqrt{(v_p-t)(v_p+11t-8tl^2)} > \sqrt{2t(v_p-t)(1-l)}$
and $t - v_p + \sqrt{2t(v_p-t)(1-l)} > 0$. Thus, both the left and right side of the equation are positive. Therefore,

$$\begin{aligned}
(t - v_p + \sqrt{(v_p-t)(v_p+11t-8tl^2)})^2 &> 8t(v_p-t)(1-l) \\
(v_p-t)(t + v_p - \sqrt{(v_p-t)(v_p+11t-8tl^2)} + 4tl(1-l)) &> 0 \\
t + v_p + 4tl(1-l) &> \sqrt{(v_p-t)(v_p+11t-8tl^2)} \\
4t((3 + 2l - 8l^3 + 4l^4)t - 2v_p(1-l)) &> 0 \\
(3 + 2l - 8l^3 + 4l^4)t &> 2v_p(1-l)
\end{aligned}$$

Notice that $l \geq \frac{2v_p-3t}{2(v_p-t)}$ is necessary for the interior solution for Founders. Hence, $2v_p(1-l) \leq t(3-2l)$, and

$$\begin{aligned}(3+2l-8l^3+4l^4)t &> t(3-2l)\\ 4l-8l^3+4l^4 &> 0\\ 1-l^2+l^3 &> 0\end{aligned}$$

And this is true for all $l \in [0.1]$. Therefore,

$$(3+2l-8l^3+4l^4)t > t(3-2l) \geq 2v_p(1-l)$$

And indeed $t(2-l) - \sqrt{2t(v_p-t)(1-l)} > \frac{t(3-2l)+v_p-\sqrt{(v_p-t)(v_p+11t-8tl^2)}}{2}$ is always true as long as an interior solution is played by both Founders and Altruists.

This concludes the proof. □

## A.4 Profit-Driven Manager

I solve the game by backward induction. Notice that in Section A.1, I have solved the indifferent user, and the pricing strategy of Firms. Hence, to show Proposition 3 I first show Lemma 3.

**Lemma 3.** *The pricing strategy of profit-driven Managers is* $p_o^M = \frac{t(3-l)-\gamma-v_p}{3}$ *and proprietary Firms is* $p_p^M = \frac{v_p-2\gamma+t(3-2l)}{3}$.

*Proof of Lemma 3.*
To show Lemma 3, recall that both Managers and Firms set prices simultaneously, and Firms' pricing strategy is given in Lemma 1. I solve for Managers' pricing strategy, then show the equilibrium pricing strategy.

Recall that Managers' profit function is given by $\pi_o^M = p_o \cdot (1-\bar{x})$, where $\bar{x}$ is given in (1).

$$\pi_o^M = p_o \frac{t+p_p-p_o-v_p}{t(2-l)-\gamma}$$

and profits are concave in price. Therefore,

$$p_o^M = \frac{p_p+t-v_p}{2}$$

accounting for $p_p = \frac{v_p + p_o - \gamma + t(1-l)}{2}$,

$$p_o^M = \frac{t(3-l) - \gamma - v_p}{3}$$
$$p_p^M = \frac{v_p - 2\gamma + t(3-2l)}{3}$$

This concludes the proof. □

Further, accounting for the pricing strategies, Restriction 1 implies that

$$t(2-l) - \gamma > 0 \tag{a.6}$$
$$v_p + t(3-2l) - 2\gamma > 0 \tag{a.7}$$
$$t(3-l) - v_p - \gamma > 0 \tag{a.8}$$

Using these pricing strategies, I now show Proposition 3.

*Proof of Proposition 3.*
I first propose a candidate equilibrium which maximises Managers' profits. I then show how such an equilibrium satisfies the Restrictions of the model.

Accounting for the pricing strategies, the indifferent user is $\bar{x} = \frac{v_p + t(3-2l) - 2\gamma}{3(t(2-l)-\gamma)}$ and Managers' profit is now

$$\pi_o^M = \frac{(t(3-l) - v_p - \gamma)^2}{9(t(2-l) - \gamma}$$
$$\frac{\partial \pi_o^M}{\partial \gamma} = \frac{(t(3-l-v_p-\gamma)(\gamma - v_p - t(1-l))}{9(t(2-l)-\gamma)^2}$$
$$\frac{\partial^2 \pi_o^M}{\partial \gamma^2} = \frac{2(v_p - t)^2}{9(t(2-l)-\gamma)^3} \ , > 0$$

First, observe that within the restrictions of the model, (a.6) and (a.8) imply that it is always profitable for Managers to coordinate the FOSS.

Second, observe that the second order condition implies that the FOSS profit is not concave in the level of coordination. Therefore, whenever $\frac{\partial \pi_o^M}{\partial \gamma} > 0$, Managers prefer the highest level of coordination. Since $t(3-l) - \gamma - v_p > 0$ and $t(2-l) - \gamma > 0$, $\frac{\partial \pi_o^M}{\partial \gamma} > 0 \iff \gamma - v_p - t(1-l) > 0$ and 0 with equality.

Suppose first that $v_p < t$. When this is true, I show that $\frac{\partial \pi_o^M}{\partial \gamma} > 0$. From $\gamma - v_p - t(1-l) >$

0,

$$\begin{aligned} \gamma - v_p - t(1-l) > 0 \text{ adding } t(2-l) - \gamma > 0, \\ t(2-l) - \gamma + \gamma - v_p - t(1-l) > 0 \\ t > v_p \end{aligned}$$

Therefore, whenever $v_p < t$, $\frac{\partial \pi_o^M}{\partial \gamma} > 0$ and Managers prefer the highest level of coordination, $\gamma^M = 1$.

Suppose next that $v_p \geq t$. When this is true, I show that $\frac{\partial \pi_o^M}{\partial M} < 0$. Suppose to a contradiction that $\gamma - v_p - t(1-l) > 0$.

$$\begin{aligned} \gamma - v_p - t(1-l) > 0 \\ t - v_p > t(2-l) - \gamma \end{aligned}$$

the left hand side is weakly negative, and the right hand side is positive. Hence, a contradiction. This implies that $\frac{\partial \pi_o^M}{\partial \gamma} < 0$, and Managers are still active, but prefers the lowest level of coordination.

I now consider the Restrictions of the model. Recall that for Restriction 1 the indifferent user and the extreme user located at 1 must have weakly positive utility, and for Restriction 2 the equations (a.6), (a.7) and (a.8) hold.

The indifferent user's utility is given by $u_o(\bar{x}) = (1-\bar{x})(\gamma - t(1-l)) - p_o = \frac{(t(3-l)-v_p-\gamma)(2\gamma - t(3-2l))}{3(t(2-l)-\gamma)}$, and the for the extreme user $u_o(1) = \frac{2(t(3-l)-v_p-\gamma)(\gamma-t)}{3(t(2-l)-\gamma))}$. For the indifferent user to receive weakly positive utility, $\gamma \geq \frac{t(3-2l)}{2}$, and for the extreme user, $\gamma \geq t$.

Because $\gamma \in [0,1]$, Restriction 1 implies that $1 \geq \frac{t(3-2l)}{2}$ or $l \geq \frac{3t-2}{2t}$ is a sufficient condition for Managers to be active and the market to be covered.

I now show when the Restrictions are satisfied by the candidate equilibrium.

Suppose first that $v_p < t$, and the candidate equilibrium is $\gamma^M = 1$. The following equations need to be satisfied.

$$\begin{aligned} 1 \geq t \\ 1 \geq \frac{t(3-2l)}{2} \\ t(2-l) > 1 \\ v_p + t(3-2l) > 2 \\ t(3-l) - v_p > 1 \end{aligned}$$

Observe that when $v_p < t$ then whenever $t(2-l) > 1$ is satisfied, both of $v_p + t(3-2l) > 2$

and $t(3-l)-v_p>1$ are satisfied. Notice that $1 \geq t$ is consistent whenever

$$\begin{aligned} t(2-l) &> 1 \geq t \\ t(2-l) &> t \\ t(1-l) &> 0 \end{aligned}$$

which is always true because $t>0$ and $l \in (0,1)$.

Finally, $1 \geq \frac{t(3-2l)}{2} \iff 2 \geq t(3-2l)$. And this is consistent whenever

$$\begin{aligned} v_p + t(3-2l) &> 2 \geq t(3-2l) \\ v_p + t(3-2l) &> t(3-2l) \\ v_p &> 0 \end{aligned}$$

And this is implicitly true, or Firms are inactive. See Etzion and Pang (2014) for details.

Therefore, whenever Restriction 1 and 2 are satisfied and $v_p < t$, Managers choose $\gamma^M = 1$.

Suppose next that $v_p \geq t$, and the candidate equilibrium is the smallest level of $\gamma$ that satisfies the following equations.

$$\begin{aligned} \gamma &\geq t \\ \gamma &\geq \frac{t(3-2l)}{2} \\ t(2-l) &> \gamma \\ v_p + t(3-2l) &> 2\gamma \\ t(3-l) - v_p &> \gamma \end{aligned}$$

Notice first that $\gamma \geq t$ and $\gamma \geq \frac{t(3-2l)}{2}$ imply that for the candidate equilibrium to satisfy Restriction 1, $\gamma^M = max\{t, t(1.5-l)\}$. Observe that $\gamma^M = t$ if $l \geq 0.5$ and $\gamma^M = t(1.5-l)$ if $l < 0.5$.

Using this, I show that the candidate equilibrium satisfies Restriction 2.

Suppose first that $l \geq 0.5$ then (a.6), (a.7) and (a.8) become

$$\begin{aligned} t(2-l) &> t \\ v_p + t(3-2l) &> 2t \\ t(3-l) - v_p &> t \end{aligned}$$

which are all satisfied because

$$\begin{aligned} t(1-l) &> 0 \\ v_p + t(1-l) &> 0 \\ t(2-l) - v_p &> 0 \end{aligned}$$

are always true because $t > 0$ and $l \in [0.5, 1)$.

Suppose next that $l < 0.5$ then (a.6), (a.7) and (a.8) become

$$\begin{aligned} t(2-l) &> t(1.5-l) \\ v_p + t(3-2l) &> t(3-2l) \\ t(3-l) - v_p &> t(1.5-l) \end{aligned}$$

which are all satisfied because

$$\begin{aligned} 0.5t &> 0 \\ v_p &> 0 \\ 1.5t - v_p &> 0 \end{aligned}$$

which are always true because $t > 0$.

Therefore, following Restrictions 1 and 2, $l \geq \frac{3t-2}{2t}$ is a sufficient condition for Managers to be active. And when $v_p \geq t$, $\gamma^M = \max\{t, t(1.5-l)\}$, when $v_p < t$, $\gamma^M = 1$.

This concludes the proof. □

## A.5 Surplus Analysis

Before embarking on the surplus analysis, I provide the expressions for consumer surplus, $CS_i$ where $i \in \{o, p, b\}$ represent the FOSS, the proprietary software and the total consumer surplus respectively.

$$\begin{aligned} CS_o &= \int_{\bar{x}}^{1} \gamma(1-\bar{x}) - t|(1-l(1-\bar{x}) - x| - p_o dx \\ &= \frac{(1-\bar{x})^2(2\gamma + 2tl(1-l) - t)}{2} - p_o(1-\bar{x}) \\ CS_p &= \frac{\bar{x}(2v_p - t\bar{x} - 2p_p)}{2} \\ CS_b &= CS_o + CS_p \end{aligned}$$

*Proof of Corollary 5.*

Evaluating $CS_o$ and $CS_p$ in the Founders equilibrium, recall that

$$\begin{aligned} p_o &= 0 \\ \bar{x} &= \frac{v_p - \gamma^* + t(1-l)}{2(t(2-l) - \gamma^*)} \\ \gamma^* &= t(2-l) - \sqrt{2t(v_p - t)(1-l)} \\ p_p^* &= \frac{v_p + p_o - \gamma + t(1-l)}{2}. \end{aligned}$$

Therefore,

$$CS_o = \frac{(t(3-2l^2) - 2\sqrt{2t(v_p-t)(1-l)})(v_p + t(1-2l) - 2\sqrt{2t(v_p-t)(1-l)})}{16t(1-l)}$$

$$\frac{\partial CS_o}{\partial l} < 0$$

and

$$\frac{\partial CS_p}{\partial l} = \frac{\sqrt{t(v_p-t)(1-l)}(\sqrt{2}v_p(1-l) + 2\sqrt{2}t(1-l)^2 - \sqrt{t(v_p-t)(1-l)})}{16t(1-l)^3} > 0$$

and

$$\begin{aligned} \pi_p &= \frac{(v - t + \sqrt{2t(v_p-t)(1-l)})^2}{4\sqrt{2t(v_p-t)(1-l)}} \\ \frac{\partial \pi_p}{\partial l} &= \frac{t(v_p-t)^2(v_p - t(3-2l))}{4(2t(v_p-t)(1-l))^{\frac{3}{2}}}. \end{aligned}$$

Recall that (a.5) implies $t(3-2l) > v_p$. Therefore $\frac{\partial \pi_p}{\partial l} < 0$.

This concludes the proof. □

*Proof of Corollary 6.*
Evaluating $CS_o$ and $CS_p$ in the Altruists equilibrium, recall that

$$\begin{aligned} p_o &= 0 \\ \bar{x} &= \frac{v_p - \gamma^* + t(1-l)}{2(t(2-l) - \gamma^*)} \\ \gamma^* &= \frac{t(3-2l) + v_p - \sqrt{(v_p-t)(11t - 8tl^2 + v_p)}}{2} \\ p_p^* &= \frac{v_p + p_o - \gamma + t(1-l)}{2} \end{aligned}$$

Then,

$$CS_o = (v_p + 2t(1-l^2) - \sqrt{(v_p-t)(v_p+11t-8tl^2)})\times \frac{(3(v_p-t)-\sqrt{(v_p-t)(v_p+11t-8tl^2)})^2}{8(v_p-t-\sqrt{(v_p-t)(v_p+11t-8tl^2)})^2}$$

$$\frac{\partial CS_o}{\partial l} \leq 0$$

and

$$CS_p = (v_p - t + \sqrt{(v_p-t)(v_p+11t-8tl^2)})\times \frac{(t^2(13-8l^2)+4v_p(\sqrt{(v_p-t)(v_p+11t-8tl^2)}) - t(v_p(9-8l^2)+\sqrt{(v_p-t)(v_p+11t-8tl^2)}))}{8(t+\sqrt{(v_p-t)(v_p+11t-8tl^2)}-v_p)^2}$$

$$\frac{\partial CS_p}{\partial l} > 0$$

and

$$\pi_p = \frac{(v_p - t + \sqrt{(v_p-t)(11t-8tl^2+v_p)})^2}{8(t-v_p+\sqrt{(v_p-t)(11t-8tl^2+v_p)})}$$

$$\frac{\partial \pi_p}{\partial l} = \frac{tl(v_p-t)(3(v_p-t)-\sqrt{(v_p-t)(11t-8tl^2+v_p)})(v_p-t+\sqrt{(v_p-t)(11t-8tl^2+v_p)})}{\sqrt{(v_p-t)(11t-8tl^2+v_p)}(t-v_p+\sqrt{(v_p-t)(11t-8tl^2+v_p)})^2}.$$

Observe that the denominator of $\frac{\partial \pi_p}{\partial l}$ is strictly positive, and $v_p - t + \sqrt{(v_p-t)(11t-8tl^2+v_p)} > 0$, and $v_p > t$. Therefore the sign of $\frac{\partial \pi_p}{\partial l}$ is the same as $3(v_p - t) - \sqrt{(v_p-t)(11t-8tl^2+v_p)}$. Suppose that $3(v_p-t) - \sqrt{(v_p-t)(11t-8tl^2+v_p)} < 0$,

$$\begin{aligned} 3(v_p-t) &< \sqrt{(v_p-t)(11t-8tl^2+v_p)} \\ 9(v_p-t)^2 &< (v_p-t)(11t-8tl^2+v_p) \\ 2v_p &< 5t - 2tl^2 \\ v_p &< t\frac{5-2l^2}{2} \end{aligned}$$

and this is the restriction required for the FOSS to be active. Therefore $\frac{\partial \pi_p}{\partial l} < 0$.

This concludes the proof. □

*Proof of Corollary 7.*

To show Corollary 7, I compare the total surplus in the market at $\gamma^A$ and $\gamma^*$.

The total surplus in the market is given as

$$\begin{aligned} TS &= CS_b + p_p \cdot \bar{x} \\ &= \frac{(1-\bar{x})^2(2\gamma - t(1-2l+2l^2)) + \bar{x}(2v_p - t\bar{x})}{2} \\ \bar{x} &= \frac{v_p - \gamma + t(1-l)}{2(t(2-l)-\gamma)} \end{aligned}$$

Then replacing $\gamma = \gamma^*$ and $\gamma = \gamma^A$, then recall that $t(2-l)-\gamma > 0$, $v + t(1-l) - \gamma > 0$, $t(3-l) - v - \gamma > 0$ and we show that $TS(\gamma^*) > TS(\gamma^A)$.

This shows that Altruists are never maximising total surplus, and Founders generate a higher level of total surplus than Altruists.

This concludes the proof. □

*Proof of Corollary 8.*
I first show that total consumer surplus ($CS_b$) decreases in $l$, then show how the profits of both proprietary Firms and profit-driven Managers decreases in $l$.

To see this, I evaluate each term in equilibrium, taking the first derivative with respect to $l$.

Recall that

$$\begin{aligned} p_o^M &= \frac{t(3-l) - \gamma - v_p}{3} \\ p_p^M &= \frac{v_p - 2\gamma + t(3-2l)}{3} \\ \bar{x} &= \frac{v_p + t(3-2l) - 2\gamma}{3(t(2-l)-\gamma)} \\ \gamma^* &= \begin{cases} 1 & \text{if } v_p < t \\ max\{t, t(1.5-l)\} & \text{if } v_p \geq t \end{cases} \end{aligned}$$

I show that the total consumer surplus in unambiguously decreasing in the permissiveness of the FOSS license.

$$CS_b = CS_o + CS_p$$

$$\frac{\partial CS_b}{\partial l} \begin{cases} < 0 & \text{when } \gamma^M = 1, \text{ subject to Restriction 1 and 2} \\ = \frac{2(t^2(-5+9l-6l^2+l^3)+tv_p(3-2l+l^2)-v_p^2)}{9t(1-l)^2} < 0 & \text{when } \gamma^M = t \text{ and } l \in [0.5, 1) \\ = -\frac{2l(3t-2v_p)^2}{9t} < 0 & \text{when } \gamma^M = t(1.5-l) \end{cases}$$

I now show how the profits of proprietary Firms decreases in the permissiveness of the FOSS license.

$$\pi_p^M = \frac{(v_p + t(3-2l) - 2\gamma)^2}{9(t(2-l)-\gamma)}$$

$$\frac{\partial \pi_p^M}{\partial l} = \begin{cases} \frac{t(t(3-2l)-v_p-2)(2+v_p-t(5-2l)))}{9(t(2-l)-1)^2} < 0 & \text{if } \gamma^M = 1 \\ -\frac{(t(1-2l)+v_p)(t(3-2l)-v_p)}{9t(1-l)^2} < 0 & \text{if } \gamma^M = t \\ 0 & \text{if } \gamma^M = t(1.5-l) \end{cases}$$

Finally I discuss how the profits of Managers change in the permissiveness of the FOSS license, and that this depends on the their choice of $\gamma$.

$$\pi_o^M = \frac{(t(3-l) - v_p - \gamma)^2}{9(t(2-l)-\gamma)}$$

$$\frac{\partial \pi_o^M}{\partial l} = \begin{cases} \frac{(t(3-l)-v_p-1)(1-v_p-t(1-l))}{9(t(2-l)-1)^2} > 0 & \text{if } \gamma^M = 1 \\ \frac{(tl-v_p)(t(2-l)-v_p)}{9t(1-l)^2} < 0 & \text{if } \gamma^M = t \\ 0 & \text{if } \gamma^M = t(1.5-l) \end{cases}$$

This concludes the proof. □

*Proof of Corollary 9.*
The total surplus evaluated at $\gamma^M$ is

$$TS = CS_b + p_p \cdot \bar{x} + p_o \cdot (1-\bar{x})$$

$$\frac{\partial TS}{\partial \gamma} = \begin{cases} \frac{t^2(7-l^2)+2(-5-2l+l^2)v_p t+2v_p^2(2+l)}{9t^2(1-l)^2} & > 0 \text{ when evaluated at } \gamma^M = t \\ \frac{t^2(33-24l^2)+8(-7+5l^2)v_p t+8v_p^2(3-2l^2)}{9t^2} & > 0 \text{ when evaluated at } \gamma^M = t(1.5-l) \end{cases}$$

Therefore, Managers do not maximise total surplus.

This concludes the proof. □

# Appendix B Proofs (Extensions)

## B.1 Founder's License Choice

*Proof of Proposition 4.*
To show this, I solve the game by backward induction. Since the last two stages of the game is identical to the main model, the proof for the indifferent user and proprietary Firms' action can be found in Lemma 2 and 1 respectively. Recall that the objective

function of Founders is

$$\begin{aligned}
\pi_o &= \gamma(1-\bar{x}) - t(l(1-\bar{x})) \\
&= (\gamma - tl)(\frac{t(3-l)-\gamma-v_p}{2(t(2-l)-\gamma)}) \\
\frac{\partial \pi_o}{\partial l} &= \frac{-t(\gamma^2 + t((6-4l+l^2)t-2v) + 2\gamma((-3+l)t+v_p))}{2(t(2-l)-\gamma)^2} \\
\frac{\partial^2 \pi_o}{\partial l^2} &= \frac{2t^2(v_p-t)(t-\gamma)}{(t(2-l)-\gamma)^3}.
\end{aligned}$$

If $\gamma \geq t$, then $\frac{\partial \pi_o}{\partial l} \leq 0$.

If $\gamma < t$, then $\frac{\partial^2 \pi_o}{\partial l^2} > 0$. And $\frac{\partial \pi_o}{\partial l} > 0 \iff v_p > \frac{t^2(6-4l+l^2)+\gamma^2-2t\gamma(3-l)}{2(t-\gamma)}$.

This concludes the proof. □

## B.2 Monopolistic Market

*Proof of Corollary 10.*
I first show that self-interested Founders and volunteering Altruists never form monopolies. I then show the conditions which allow profit-driven Managers to form a monopoly, and proprietary Firms to form a monopoly.

To see that Founders and Altruists never form monopolies, observe that $\bar{x} = \frac{v_p-\gamma+t(1-l)}{2(t(2-l)-\gamma)}$. For $\bar{x} \leq 0$, $v_p + t(1-l) - \gamma \leq 0$. This means $\gamma \geq v_p + t(1-l)$ is necessary for the FOSS to form a monopolist. At the interior solution for both Founders and Altruists, this is not true.

To see when Managers form a monopoly, observe that $\bar{x} = \frac{v_p+t(1-2l)}{t(4-3l)-\gamma}$. For $\bar{x} \leq 0 \iff v_p \leq t(2l-1)$.

To see when Firms form a monopoly, observe that $\bar{x} \geq 1 \iff v_p \geq t(3-l) - \gamma$. By definition of monopoly, $\gamma = 0$. Therefore Firms form a monopolist whenever $v_p \geq t(3-l)$.

This concludes the proof.

□

## B.3 Skilled Users

*Proof of Proposition 5.*
This game is identical to the main model, with the following change: users have heterogeneous skill level, and those located at 1 have a higher skill level than those located at 0. I solve this game by backward induction to find $\gamma^S$, the optimal unique contributions for Founders.

First, I look at the decision made by the marginal user.

$$v_p - p_p - t\bar{x} = \frac{(1-\bar{x})(\alpha(1+\bar{x})(\gamma+tl)-2t)}{2}$$

From this, we can find partial derivatives for $\bar{x}$ with respect to $p_p$ and $\gamma$.

$$\begin{aligned}\frac{\partial \bar{x}}{\partial p_p} &= \frac{1}{\bar{x}\alpha(\gamma+tl)-2t} \\ \frac{\partial \bar{x}}{\partial \gamma} &= \frac{\alpha(1-\bar{x}^2)}{2(\bar{x}\alpha(\gamma+tl)-2t)}\end{aligned}$$

Solving explicitly for $\bar{x}$,

$$\bar{x} = \frac{2t \pm \sqrt{4t^2+\alpha^2(\gamma+tl)^2-2\alpha(\gamma+tl)(v+t-p)}}{\alpha(\gamma+tl)}$$

Moving to the next stage, we look at the decision made by Firms. Notice that for any real interior solution, this means that $\frac{\partial \bar{x}}{\partial p} < 0$. This implies that $2t > \bar{x}\alpha(\gamma+tl)$. And additionally that we reject the positive of $\bar{x}$, and $\bar{x} = \frac{2t-\sqrt{4t^2+\alpha^2(\gamma+tl)^2-2\alpha(\gamma+tl)(v+t-p)}}{\alpha(\gamma+tl)}$. This means for positive demand, $p < v + t - \frac{\alpha(\gamma+tl)}{2}$ and for a real solution to $\bar{x}$, $p \geq v + t - \frac{4t^2+\alpha^2(\gamma+tl)^2}{4\alpha(\gamma+tl)}$. Hence, for a duopoly it must be that $2t > \alpha(\gamma+tl)$.

Solving for the optimal price,

$$\begin{aligned}\pi_p &= p\bar{x} \\ \frac{\partial \pi_p}{\partial p} &= \bar{x} + \frac{\partial \bar{x}}{\partial p}p\end{aligned}$$

For a solution, $\frac{\partial \pi_p}{\partial p} = 0$ and this implies that $p_p = -\frac{\bar{x}}{\frac{\partial \bar{x}}{\partial p_p}} = \bar{x}(2t - \bar{x}\alpha(\gamma+tl))$.

$$\begin{aligned}\frac{\partial^2 \pi_p}{\partial p_p^2} &= 2\frac{\partial \bar{x}}{\partial p} + p\frac{\partial^2 \bar{x}}{\partial p^2} \\ &= \frac{3\bar{x}\alpha(\gamma+tl)-4t}{(2t-\bar{x}\alpha(\gamma+tl))^2} < 0\end{aligned}$$

For an interior solution, $3\bar{x}\alpha(\gamma+tl) - 4t < 0$ or $\bar{x} < \frac{4t}{3\alpha(\gamma+tl)}$ or $\gamma < t(\frac{4}{3\bar{x}\alpha} - l)$.

These two conditions together imply that $2t > \alpha(\gamma+tl)$ or that $\alpha < \frac{2t}{\gamma+tl}$.

Together, the three conditions $\alpha < \frac{4t}{3\bar{x}(\gamma+tl)}$, $\alpha < \frac{2t}{\bar{x}(\gamma+tl)}$ and $\alpha < \frac{2t}{\gamma+tl}$ suggest that an interior solution is only possible if $\alpha$ is sufficiently small, bound above by $\frac{2t}{\gamma+tl}$ if $\bar{x} \leq \frac{2}{3}$ and $\frac{4t}{3\bar{x}(\gamma+tl)}$ if $\bar{x} > \frac{2}{3}$. In all other cases, there will be zero demand for the proprietary software. That is, if users are exceptionally skilled, they simply create a unique product

on their own, driving out proprietary Firms. Given this, I now turn to the Founders' choice. Observe that $\alpha < \frac{2t}{\bar{x}(\gamma+tl)}$ is the least restrictive condition. Also notice that rewriting $\bar{x} < \frac{4t}{3\alpha(\gamma+tl)}$ and $\bar{x} < \frac{2t}{\alpha(\gamma+tl)}$ suggests that $\bar{x}$ decreases in $\alpha$.

Moving to the first stage, the decision by Founders is

$$\pi_o = \frac{\alpha(\gamma - tl)(1 - \bar{x}^2)}{2}$$

And subject to $\gamma \leq 1$, letting $\lambda$ be the shadow price,

$$\begin{aligned}\frac{\partial \pi_o}{\partial \gamma} &= \frac{\alpha(1 - \bar{x}^2 - 2(\gamma - tl)\bar{x}\frac{\partial \bar{x}}{\partial \gamma})}{2} - \lambda \\ &= \frac{\alpha t(1 - \bar{x}\alpha l)(1 - \bar{x}^2)}{2t - \bar{x}\alpha(\gamma + tl)} - \lambda\end{aligned}$$

$\lambda \geq 0$, $\lambda(1 - \gamma) = 0$, $1 - \gamma \geq 0$.

Notice that for $\frac{\alpha t(1-\bar{x}\alpha l)(1-\bar{x}^2)}{2t-\bar{x}\alpha(\gamma+tl)} - \lambda = 0$, either $\bar{x} = \frac{1}{\alpha l}$ and $\lambda = 0$ or $\lambda = \frac{\alpha t(1-\bar{x}\alpha l)(1-\bar{x}^2)}{2t-\bar{x}\alpha(\gamma+tl)}$. In the first case, $\lambda = 0$ implies that this solution can be true for any $\gamma$. In the second case, $\lambda > 0$ implies that $\gamma = 1$.

Observe that the coordinator is only active if $\gamma > tl$. Suppose that $\gamma \neq 1$. Then $\frac{\partial \pi_o}{\partial \gamma} \geq 0 \iff (1 - \bar{x}\alpha l) \geq 0$, and 0 only with equality This means should there be an interior solution for the FOSS where $\bar{x} = \frac{1}{\alpha l}$.

However, I show that $\bar{x} = \frac{1}{\alpha l}$ leads to a contradiction between an active FOSS $\gamma > tl$ and an active Firm $2t - \bar{x}\alpha(\gamma + tl) = 2t - \frac{\alpha(\gamma+tl)}{\alpha l} > 0 \iff tl > \gamma$. Hence, a duopoly cannot exist if $\bar{x} = \frac{1}{\alpha l}$.

Therefore, the optimal solution for Founders is $\gamma^S = 1$.

I conclude that skilled self-interested Founders choose to develop a FOSS if $1 > tl$, and they prefer that contributions of each developer is unique, $\gamma^S = 1$.

This concludes the proof. □

## B.4 Non-User Developers

*Proof of Proposition 6.*
I use backward induction to solve for the equilibrium. This is done in the following steps: (1) determine the marginal user indifferent between consumption of the proprietary software and the FOSS; (2) determine the marginal developer indifferent between other leisure and the development of the FOSS; (3) solve for the price strategy of the Firm; (4) solve for the optimal level of coordination for developers of the FOSS ($\gamma^D$).

In deciding which software to use, users compare between the following utilities, $u_p = v_p - p_p - tx$ and $u_o = \gamma(1-\bar{o}) - t|(1 - l(1-\bar{o}) - x)|$. The demand here arises from the uniform distribution. The indifferent user being $\bar{x}$, and $\bar{o}$ the indifferent developer.

$$\begin{aligned} v_p - p_p - t\bar{x} &= (1-\bar{o})(\gamma + tl) - t(1-\bar{x}) \\ \bar{x} &= \frac{v_p - p_p - (1-\bar{o})(\gamma + tl) + t}{2t} \end{aligned}$$

From Restriction 1, $\bar{x}$ gives the demand for the proprietary software, and $(1-\bar{x})$ the demand of the FOSS. I turn now to the decision of the developers.

Developers decide between contributing to the FOSS and receiving a utility of $w_o = s_o - k|L_o - o| = \beta(1-\bar{x}) - k|(1 - l(1-\bar{o}) - o)|$. Thus the indifferent developer is

$$\begin{aligned} \beta(1-\bar{x}) - k(1 - l(1-\bar{o}) - \bar{o}) &= 0 \\ \bar{o} &= \frac{\beta(\bar{x}-1) + k(1-l)}{k(1-l)} \\ &= 1 - \frac{\beta(1-\bar{x})}{k(1-l)} \end{aligned}$$

This means that

$$\begin{aligned} \bar{x} &= \frac{k(1-l)(v+t-p) - \beta(\gamma + tl)}{2tk(1-l) - \beta(\gamma + tl)} \\ \bar{o} &= \frac{2tk(1-l) + \beta(v - p - t(1-l) - \gamma)}{2tk(1-l) - \beta(\gamma + tl)} \end{aligned}$$

We will need $k > \frac{\beta(\gamma+tl)}{2t(1-l)}$ for concavity of the profit function of Firms.

Now we turn to the decision of Firms.

$$\begin{aligned} \pi_p &= p_p \bar{x} \\ p_p &= \frac{1}{2}(v_p + t - \frac{\beta(\gamma + tl)}{k(1-l)}) \\ \bar{x} &= \frac{k(1-l)(v+t) - \beta(\gamma + tl)}{2(2tk(1-l) - \beta(\gamma + tl))} \\ \bar{o} &= \frac{4tk^2(1-l)^2 + \beta^2(\gamma + tl) + k\beta(1-l)(v_p + 2tl - 3t - 2\gamma)}{2k(1-l)(2tk(1-l) - \beta(\gamma + tl))} \\ \pi_p &= \frac{((v+t)k(1-l) - \beta(\gamma + tl))^2}{4k(1-l)(2tk(1-l) - \beta(\gamma + tl))} \end{aligned}$$

Finally, we turn our attention to the coordinator of the FOSS. Recall that Founders are

selfish, and is only motivated by maximising its individual utility.

$$\begin{aligned}
\pi_o &= \beta(1-\bar{x}) - kl(1-\bar{o}) \\
&= \frac{\beta(\beta(2l-1)(\gamma+tl) + k(1-l)(t(3-6l+4l^2)+v_p(2l-1)))}{2(1-l)(2tk(1-l)-\beta(\gamma+lt))} \\
\frac{\partial \pi_o}{\partial \gamma} &= \frac{\beta^2 k(4tl^2+(2l-1)(v_p-t))}{2(2tk(1-l)-\beta(\gamma+tl))^2)} \\
\frac{\partial^2 \pi_o}{\partial \gamma^2} &= \frac{\beta^3 k(4tl^2+(2l-1)(v_p-t))}{(2tk(1-l)-\beta(\gamma+tl))^3}
\end{aligned}$$

Observe that both $\frac{\partial \pi_o}{\partial \gamma}$ and $\frac{\partial^2 \pi_o}{\partial \gamma^2} > 0 \iff$ both $2l-1>0$ and $v_p - t > 0$ simultaneously. Hence, the FOSs is only active if there is a sufficiently permissive license when the proprietary software has a high quality, and a sufficiently restrictive license when the proprietary software has a low quality. In either case, Founders select $\gamma^D = 1$ preferring that contributions by each developer is unique.

Notice that $\beta$ does not influence the outcome of this decision.

This concludes the proof.

□