

# Peer production, division of labor, and project modularity

Daniel H. Neilson\*

15 January 2009

## Abstract

Peer production represents a mode of coordination of productive activity that is distinct from market- or firm-based coordination. Large projects must be divided, the parts assigned to many collaborators and reassembled into a functioning whole. A model of project modularity sheds some light on the division of labor in this context. A hierarchical version of the model shows that the optimal division of a project is one that distributes work equally between levels of the hierarchy. If participants are organized into hierarchical production teams, variations in their organizational structure have consequences for productivity, and different choices of structure are appropriate under different circumstances.

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Modularity is essential to peer production</b>	<b>4</b>
<b>3</b>	<b>A model shows the benefits of modularity</b>	<b>11</b>
3.1	Simon's model is a good place to start . . . . .	12
3.2	The model does not assume a single participant . . . . .	20
3.3	Modularity has costs, too . . . . .	22
<b>4</b>	<b>Hierarchical production, multiple agents: simulation model</b>	<b>29</b>
4.1	Details of the simulation model . . . . .	29
4.2	Comparison to the analytical model . . . . .	31
4.3	Anticipating some objections to the model . . . . .	34
<b>5</b>	<b>The shape of the hierarchy matters</b>	<b>38</b>
<b>6</b>	<b>Conclusions</b>	<b>42</b>
6.1	Lessons for designers of open-source systems . . . . .	42
6.2	Consequences for peer-production researchers . . . . .	44
6.3	Endogeneity of project design . . . . .	44
6.4	Methodological choices . . . . .	46
6.5	Finally . . . . .	47

---

\*dhn2102@columbia.edu. Ph.D. student, Department of Economics, Columbia University.

---

# 1 Introduction

The division of labor is central to the economic analysis of production. At least since Smith (1776) made his famous observations of laborers at an pin factory, the potential has been recognized for a team working in coordination to achieve a level of productivity not attainable by the same number of individuals each working in isolation. By allowing each worker to focus on a single task, in Smith's case a particular step in the manufacture of pins, such an efficiency at each stage can be achieved that the output of the group is many times larger than that of the same number of individuals working independently.

Smith realized that incurring the benefits of specialization and division of labor was not simply a question of ever-finer division of the process into parts. He notes, for example, the benefits of “a proper division and combination” of the steps in the manufacture of pins. Not only must the work be divided, it must be done so logically, and in such a way that the parts can be combined into the finished whole. Tasks could be assigned one to each worker, or one or two could be handled by a single individual. The tasks themselves, however, should be coherent, and further subdividing these tasks and giving parts of them to different individuals would sacrifice productivity. Ever-finer subdivision, then, is not by itself what gives rise to the productivity gains of the division of labor. The nature of the division, and the logic of the assignment of tasks to individuals, matter as well.

Division of labor in Smith's time was enabled in part by the technological innovations associated with the Industrial Revolution. New types of capital allowed, and perhaps necessitated, innovations in the organization of production to make use of them. A technological innovation in our own time, a radically improved ability to communicate all types of information between individuals, could equally be said to allow a new mode of production that takes advantage of it. This mode we refer to as *peer production*. The term will be defined carefully below, but the reader can safely keep two relatively familiar examples in mind as a guide. The first is Wikipedia<sup>1</sup>, “the free encyclopedia that anyone can edit”. Wikipedia must be considered a successful peer-production project—with millions of articles on nearly any imaginable topic, it is the ninth most popular English-language site on the web<sup>2</sup> and is widely known as a first reference for situations where authoritative knowledge is not required. The second example is open-source software. Open-source software constitutes a class of computer programs whose source code, the underlying instructions that make it work, is made freely

---

<sup>1</sup><http://www.wikipedia.org>

<sup>2</sup>See <http://www.alexa.com>.

---

available to anyone who wants it—it is given away at no cost. Moreover, knowledgeable individuals can take this source code and modify it to add features or fix bugs, and can share these changes with other users. Successful open-source projects include the web browser Firefox, the web server Apache, the Linux operating system kernel and many of the applications that run on Linux systems. There are hundreds of thousands of open-source projects, covering the entire domain of computing applications.<sup>3</sup>

With these examples in mind, I offer a general characterization of peer production, which comes from Benkler (2006):

radically decentralized, collaborative, and nonproprietary; based on sharing resources and outputs among widely distributed, loosely connected individuals who cooperate with each other without relying on either market signals or managerial commands. (Benkler, 2006, p. 60)

The features that Benkler mentions present a formidable coordination problem. In the absence of prices or managers, a decentralized group of individuals working cooperatively is able to share its resources and work product. As suggested above, the issue of division of labor must play a central role in this collaborative effort. How is this division achieved, and how is a functioning product derived from the parts completed by loosely connected individuals?

Its economic relevance means that peer production cannot be characterized simply as a social phenomenon. Increasingly, peer-production solutions compete with and sometimes replace comparable products produced by traditional for-profit firms. Several examples are well known. Apache, for example, is an open-source software package that runs on some two-thirds of all web servers. Its market share is significant, though that share can be measured only by counting installations, not by measuring sales—there are no sales to measure! The web browser Firefox is another well-known open-source software project. It has about a 20% of the browser market, competing primarily against the main commercial alternative, Microsoft's Internet Explorer. Beyond software, Wikipedia can be thought of as competing with traditional encyclopedias such as Britannica, and to the extent that users look to it rather than to those for-profit alternatives, it carries economic costs for them.

What is more, significant resources are devoted to the pursuit of peer-production projects, and derivative commercial products rely on peer-produced frameworks to provide additional value. The economic significance of peer production, then, cannot be ignored. While much energy has been devoted to understanding the operation of markets, as well as to the working of firms, the coordination

---

<sup>3</sup>As of this writing, over 170,000 projects were listed on <http://www.sourceforge.net>.

---

mechanisms of peer production are not as well understood.

This paper makes use of a simple theoretical model, approached first analytically and then using simulation, to make several specific points about modular peer production. First, in the simplest of frameworks, I show the benefits of choosing more modularized over less modularized project designs. I extend this in a natural way to a setting that incorporates the costs to modularization as well as the benefits, and show that the optimal modularization of a project with multiple stages of production is one that shares work evenly between the stages. This is the main result of the analytical portion of the paper. Implementing a simulation of the model, I consider collaborative projects and the organization of participants into hierarchical production teams. I show that different shapes of hierarchy are better choices under different circumstances.

The remainder of the paper is organized as follows. In Section 2, I discuss the phenomenon of peer production in greater depth, highlighting some key general features. In Section 3, I present a model that captures features of peer-production systems as discussed above, and discuss some analytical results. In Section 4, I extend the analytical model by implementing it in a simulation, which permits the addition of certain parameters. In Section 5, I present simulation results for several variations of the model. I conclude in Section 6 by considering some limitations of my approach and methodological choices; the relationship of this work to the work of others, and some lessons for those who research or participate in open-source projects.

## 2 Modularity is essential to peer production

Peer production, in particular open-source software, seems to represent a specific, recurring mode of collaboration, by which individuals, driven by diverse motives and with varying resources to contribute, collaborate on a huge variety of software and other projects. They receive no wages and are not bound into traditional manager–worker relationships. They do not sell their product and are thus not responsive to prices. Nonetheless, researchers and practitioners alike see enough consistency and common features across peer-production efforts to group them together, as economists would group firms or markets (Fogel, 2005; Benkler, 2006). As peer production has grown, certain norms, institutional designs, and infrastructure have become standard, formalizing peer production as a means of achieving coordination in the interests of production. The aim of this paper is to understand theoretically how this coordination can generate successful project outcomes. To fix ideas, it is useful

---

to present here two scenarios that exemplify more general processes to be studied in greater rigor below.

A Wikipedia contributor, curious and somewhat knowledgeable about the idea of crowdsourcing, refers to the Wikipedia page on the subject, which has some helpful information but which is not complete. He notices that it also contains some errors. He clicks on the “edit” tab of the page and modifies the source of the page to correct the errors. Along the way, he notices a passage that seems to reflect an opinion rather than a fact, which is inappropriate for an encyclopedia article. Lacking the inclination to deal with the issue at the moment, he marks the page as lacking a neutral point of view, saves his changes, leaving a brief descriptive message about what he did to the page, and goes on with his day.

Meanwhile, an economist, writing up some recent results for publication, notices that the open-source software she uses for producing figures lacks a certain feature that would allow her to most clearly present her results. A competent programmer, she hunts through the source code to find the relevant files, and realizing that implementing the feature is only a few minutes’ work, she does so. Testing it on her own data, she corrects a few bugs, checks that the code is well organized and commented clearly, and submits the change to SourceForge, the website that the software’s authors use to collaborate on the project.

First, note that both contributors have ultimate goals beyond their contribution to their respective projects. The Wikipedia contributor is simply curious and happy to share his knowledge, and believes in Wikipedia’s goal of maintaining a certain tone and style. The open-source contributor is a researcher, a knowledgeable user of open-source software with a need for a particular feature. Having implemented it, expecting to gain from the progress it enables her to make in her research, she sees no harm in submitting the change to the community of users, some of whom may also benefit from it. Perhaps she derives satisfaction from participation in the community, or from seeing her code used by others. Such motivations, and this style of user-driven innovation, are not unique to these domains (von Hippel, 1988, 2005). The salient feature of peer production is its ability, in many successful cases, to harness and aggregate the activity of these numerous and diverse potential contributors.

Some authors have sought to understand the motivations driving participation in open-source projects, typically along lines of thought established in labor economics (Lerner and Tirole, 2002).

---

<sup>4</sup> More plausibly, Hunter and Quiggin (2008) cites “altruism, self-expression, advocacy of particular political or social views, the display of technical expertise, social interaction, and so on” (pp. 205–206). The length of this list suggests the key point—namely, the fact that successful projects appear to work *regardless* of the motivations of participants. Wikipedia is a case in point. It has been shown that the activity distribution of contributors to Wikipedia is highly skewed, meaning that a small number of individuals are very active, and a large number are very rarely active (Wilkinson, 2008). Those who make hundreds of contributions per week certainly have different motivations than do those who work only on a page or two, or those who only clean up typos or fix broken links. In this paper, the feature of Wikipedia I wish to study is not why individuals should want to contribute, but rather how, in the presence of this diversity of incentives, they are collectively able to produce a coherent product.

Most participants in open-source projects do so for no direct monetary reward.<sup>5</sup> Participants must be earning a living from some other source and participating in open-source projects in their spare time; or earning money indirectly from their contributions, for example by selling a derivative product; or expecting to eventually benefit from their contributions, for example by exploiting their open-source reputation in the labor market (Lerner et al., 2006). Ignoring for the moment issues of motivation, one can imagine contributions as happening randomly, as I do explicitly in the model presented below. A successful project must be designed to collect these random part-time contributions and aggregate them into a functioning whole. Without a good system for doing so, effort undertaken by potential participants would be lost, with no way of becoming part of a larger peer-production project.

The two scenarios highlight a second key element of the decentralized production process under study. If, as is the case with open-source software, many people intend to work together to produce a good in a decentralized manner, it is clear that the division of labor will play a role in an understanding of their activity. Whether the collaboration consists of two individuals or thousands, a way must be found for each to contribute his or her part, and for these parts to be aggregated into a coherent whole. The Wikipedia author was able to bring his knowledge of a single topic into a useful relation with the knowledge of thousands of others by working on a single encyclopedia

---

<sup>4</sup> O’Mahony (2003), meanwhile, argues that licensing and behavioral norms allow projects to maintain control over their work, which could support other analyses of contributor motivations.

<sup>5</sup>It remains possible, however, that those participants who contribute the most are the ones who *are* somehow remunerated for their contributions. If this is the case, then a greater share of *contributions* than of *contributors* would be remunerated. Work in this direction is undertaken by Lerner et al. (2006).

---

page. The open-source programmer was able to coordinate her programming ability with that of the other programmers on her project by solving a single problem that she identified. In both cases, the individual efforts in isolation would be of little value, but as part of collective production efforts take on greater value.

The feature of peer production that allows collective production to arise from the decentralized contributions of a potentially large number of individuals is modularity, as noted by many authors. By this it is meant that the project can be divided into functionally distinct components. Modularity has been defined variously by different authors, so it is worth being precise about the concept here. Any large project can be divided into smaller ones. When all of the smaller ones are complete, the project is complete. The parts of a large project are likely to have interdependencies, however—steps that must be taken in order or simultaneously. *Modularity* is a quality of a particular choice of subdivision of a large project into tasks. A subdivision is said to be more modular when the tasks are not very interdependent, meaning that they can be completed in any order or simultaneously; it is less modular when the tasks are tightly interconnected, requiring that their completion be carefully choreographed.<sup>6</sup>

In the Wikipedia scenario, the author was able to modify the article, contributing his knowledge on a single subject, without coordinating directly with any other participant. The coordination was handled indirectly by the design of Wikipedia's system itself, for example by the manner in which a page could be marked as having a particular problem. Likewise, the open-source contributor was able to implement her feature without coordinating directly with other programmers. By obeying certain accepted norms, such as commenting her code, her contribution could become part of the collective effort.

Modularity is central to peer production. Much of the institutional design of peer production—the content-management system underlying Wikipedia, the various software-development tools used in open-source software—serves to enable and manage precisely this feature. Without modularity, the coordination task of peer production becomes much more difficult. In a typical microeconomic analysis of a firm, wages can be used to induce coordination among workers. In a market, likewise, price signals coordinate the activities of market participants (Mas-Colell et al., 1995). In the absence

---

<sup>6</sup> A metaphor from graph theory may also be helpful. Suppose that a project is made up of indivisible steps, which constitute the nodes of a graph. These nodes are linked by edges representing their interdependencies. Specifically, two nodes are joined by a (directed) edge if the successful completion of one requires the successful completion of the other. A subdivision of such a project into tasks is a partition of the nodes into disjoint subsets. Modularity is then a measure of how many edges cross from one set of the partition into another. A very modular project can be divided into tasks so that no steps from one set are dependent on steps from another.

of these coordinating mechanisms, it is modular project design that allows production to occur.<sup>7</sup>

The subject of modularity has been treated by other authors. Langlois (2002) studies modularity in firms by looking at property rights and control. Kogut and Metiu (2001) discusses code modularity in the context of peer-production project governance structures. Benkler (2006, pp. 100–103), writing specifically about peer production, identifies modularity as a factor in successful projects, and describes the failure of some non-modular attempts.<sup>8</sup>

The meaning and significance of modularity is examined using a model in the sections that follow. First, three examples from the brief history of peer production support the claim that small modular contributions are central to the projects' success. Wikipedia as it exists today grew out of an earlier attempt to create a free online encyclopedia, known as Nupedia. Nupedia was based around extensive peer review, designed to achieve article quality comparable to that of print encyclopedias. For the same reason, articles were to be written by experts, and a substantial initial contribution was to be made by a single qualified individual. Viewed in terms of project modularity, these initial contributions were comparatively large—not the work of a few minutes or even hours. Jimmy Wales, founder of Nupedia, proposed the creation of a companion wiki that anyone could edit:

“Wiki,” pronounced \wee'-kee\, derives from a Polynesian word, “wikiwiki,” but what it means is a VERY open, VERY publicly-editable series of web pages. For example, I can start a page called EpistemicCircularity and write anything I want in it. Anyone else (yes, absolutely anyone else) can come along and make absolutely any changes to it that he wants to. (The editing interface is very simple; anyone intelligent enough to write or edit a Nupedia article will be able to figure it out without any trouble.) On the page I create, I can link to any other pages, and of course anyone can link to mine. The project is billed and pursued as a public resource. There are a few announced suggestions or rules. [...]

A Nupedia wiki would instruct users to try to make their entries resemble encyclopedia articles, but the usual wiki sort of banter would be permitted. This would make things more interesting to many users, who could \*instantly\* create, edit, and comment on articles. If a wiki article got to a high level it could be put into the regular Nupedia editorial process. (Wales, 2001)

Making contributions easier was seen as a way of accelerating the development of Nupedia pages.

This wiki became Wikipedia. The open-contribution model proved to lead to much higher partici-

<sup>7</sup> An anecdote of Brooks (1975), retold in Langlois (2002), is instructive, though it does not deal with peer production *per se*. Workers on IBM's OS/360 operating system were asked to coordinate with by each maintaining a set of documentation for the entire project, including parts he or she was not working on directly. Within six months, however, the documentation had grown to a stack of paper five feet high, with 150 pages, or about 2 inches, of changes to be inserted daily. Non-modular project design makes it difficult for individuals to contribute.

<sup>8</sup> A distinction is made by some authors between the assignment of tasks to individuals and the self-containedness of those tasks, calling the first “modularity” and the second “decomposability” (Langlois, 2002). No such distinction is made in this paper.

pation than did the peer-review model, and community editing proved capable of generating articles of sufficient quality for Wikipedia to attract users.

Though Nupedia was abandoned, other endeavors still attempt to correct what is seen as a lack of credibility in Wikipedia by recruiting academic experts to write articles and implementing a peer-review system to maintain their quality. Chief among these are Scholarpedia<sup>9</sup> and Citizendium<sup>10</sup>. Having experts produce articles, rather than relying on volunteer contributions with no vetting of expertise, significantly increases the contribution size required of articles' initial authors. Whereas a Wikipedia article is allowed to accumulate over time as the result of small contributions, an expert-written Scholarpedia article is a significant work product, on a scale comparable to that of a journal article. There seems to be little hope of these expert-written articles achieving the success of Wikipedia, at least in terms of breadth and variety of content, and as of this writing, they carry only a tiny fraction of Wikipedia's two million English-language articles. Admittedly these new projects have different aims, namely the production of authoritative references on a narrower range of subjects. Moreover, Wikipedia's success has itself changed society's perception of free encyclopedias. There is no reason, then, to assume that these efforts will meet the same fate as Nupedia. But it seems clear that the large contribution size of expert-driven encyclopedias is a key limit to their scale.

A second example of the exploitation of highly granular collaborative project design can be found in the reCAPTCHA<sup>11</sup> (von Ahn et al., 2008), a variation on the CAPTCHA<sup>12</sup>. CAPTCHAs have become familiar to users of the internet since 2000, when they were first developed. A version of a Turing Test (i.e., a test to distinguish between humans and computers), the CAPTCHA seeks to prevent the automated registration of e-mail accounts and other online services. Typically, the distorted image of a word or series of letters is presented to the user, who must correctly recognize the word to continue in the registration process. An example of the type of test a user would be asked to perform can be found in Figure 1. Reading distorted text is of little difficulty to humans, but poses a greater challenge for computer programs.

---

<sup>9</sup> <http://www.scholarpedia.org>, retrieved 4 Sept. 2008

<sup>10</sup> <http://www.citizendium.org>, retrieved 4 Sept. 2008

<sup>11</sup> <http://recaptcha.net>, retrieved on 13 January 2009

<sup>12</sup> Completely Automated Public Turing test to tell Computers and Humans Apart

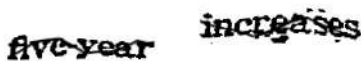


Figure 1: An example of a reCAPTCHA. A human has little difficulty discerning that the two words are “five-year” and “increases”, but the task is much more difficult for a computer.

As CAPTCHAs became commonplace, it was recognized that, in the millions of tiny deciphering problems completed daily, a great deal of effort was being wasted. The reCAPTCHA seeks to capture this effort by offering, alongside a randomly-chosen word, a scanned image from a text archive, for example the back issues of *The New York Times*. Performing automated optical character recognition (OCR) on such texts is difficult and error-prone, but, like the standard CAPTCHA problem, is not hard for humans. The reCAPTCHA asks users to perform this OCR task alongside a standard CAPTCHA. For example, in Figure 1, one of the two words would be a known word, and the other would be an unknown word from an old issue of the *Times*. The user solves both, which both proves that she is human and provides the digitized text for the unknown word. The system then implements a system for aggregating the myriad interpreted images back into complete texts. What would be a gargantuan task, digitizing large amounts of scanned text, is made possible by a clever division and reassembly of individual efforts. The effort is made without concern for or, for the most part, knowledge of the ultimate purpose; rather individuals have some other motivation for participating. Key to the success of reCAPTCHA is the very small contribution size—it is the work of only a second to identify and submit a word—and the design of the aggregation system, which permits those contributions to be put to use.

Amazon’s Mechanical Turk service<sup>13</sup> is the third and final example of the exploitation of highly modular project design. Dubbed “artificial artificial intelligence”, Mechanical Turk provides large-scale, routinized, repetitive services for its clients. A typical example is image tagging, that is, labeling images with text describing their contents. Computer programs are not yet able to reliably perform this service, so Mechanical Turk provides it artificially—by hiring out individuals to do it manually. For humans, image tagging is a relatively straightforward task. Participants are paid a small piece rate for their contributions. Mechanical Turk has been successful for a certain type of small, discrete, repetitive task—that is, for projects with very small modular units of contribution.

<sup>13</sup> <https://www.mturk.com/mturk/welcome>, retrieved 13 January 2009.

Wikipedia, reCAPTCHA and Mechanical Turk exemplify modular production. Each provides a valuable service—general reference, text digitization, or image tagging—by dividing complex projects into small but coherent units, assigning these out to be solved, and reassembling the completed units and providing the results back to the client. In each case, the division of the project into small units of contribution, and their reassembly, are made possible in part by the nature of the task, and in part through the use of clever systems for coordination.

### **3 A model shows the benefits of modularity**

As suggested above, and as indicated by many authors, a project’s modular design is key to its being amenable to peer production. If it is to be tackled by a large group of diverse participants who may not be in constant communication and who are not under strict managerial control, a project must be divisible in such a way as to permit contributors to work individually on many components, and then to assemble their parts into a working whole with a minimum of adjustment to their individual components. A design possessing this feature is said to *modular* or *loosely coupled*. In this section, I examine features of modular design using an analytical model. I begin with a simple single-participant model of a modular production process; I then consider its adaptation to a multiple-person production team. I then consider a hierarchical version of the model, again first with one contributor and then with many.

There are several key results in this section. In the simplest version of the model, I show the most basic consequence of a modular project design, namely profound resilience to the factors that tend to inhibit production. The logical conclusion of this, however, is counterintuitive, and in Section 3.3, I adapt the model in a straightforward way to reflect both the costs and the benefits of modular design. The main result that follows is that, in a hierarchical production process, work should be shared equally between the various stages of production, in a precise sense. Moreover, when the first and second stages of production occur in different environments, this sharing must be adapted in an intuitive way. Along the way, I give some interpretation of these results in various contexts, and set the stage for the more complex version of the model that follows in Section 4.

### 3.1 Simon’s model is a good place to start

To see concretely the consequences of modular design for project outcome, I consider a version of a model by Simon (1962). Simon considers two watchmakers, Tempus and Hora, working on watches consisting of the same number of parts, and capable of adding pieces to the assembly at the same rate. The two watchmakers work in an environment of interruptions, so that during the addition of each part there is some chance that they will fail and the part they are working on will have to be restarted. Here Tempus and Hora differ. Tempus’s design is monolithic, which is to say that at each interruption, the entire assembly comes apart. Hora’s design, by contrast, is such that at each interruption, only the current module, a small subset of the assembly, is lost.

Interruptions in this model can be thought of as representing all frustrating influences on the aggregation of contributions into the project as a whole. One can think of trying to write an e-mail and being repeatedly interrupted by phone calls. After each interruption, the entire e-mail must be re-read to recall one’s place. Perhaps more realistically, one could think of trying to write a paper and being continually interrupted by e-mail.<sup>14</sup> Interpreted in the context of peer production, contributions—be they bug fixes, new features, or improvements to Wikipedia articles—are prevented from being incorporated into the project for a variety of reasons, which I sum up as the interruption probability.

Hora’s design, in which only part of the work is lost at each interruption, I refer to as modular. The particular version of modularity represented in this model is a simple one—all of the steps *within* each module are interdependent and must be completed without interruption. An interruption requires that the module be started from scratch. Modules themselves are not interdependent, so that an interruption does not affect already-completed modules. To stretch terminology a bit, a design along these lines can be said to be more or less modular, which is to say that the number of parts in each module is smaller or greater, respectively.

The differences between Tempus’s and Hora’s project designs—specifically, in the modularity of their designs—have dramatic consequences for the two watchmakers’ overall productivities. Although Tempus and Hora add pieces to their assemblies at the same rate, the speed at which they are able to produce finished work can differ dramatically. We refer to this last quantity, the number of pieces in finished products per piece that had to be added to an assembly (and perhaps subsequently

---

<sup>14</sup> González and Mark (2005) document ethnographically the effects of multi-tasking in information workers. Much of this multi-tasking originates from interruptions, and requires workers to constantly attend to their open tasks and to manage transitions between them.

lost due to an interruption), as *effective productivity*. Varying the frequency of interruption in this model shows the origins of this productivity difference. At one extreme, where there are never any interruptions, Tempus and Hora are able to complete their projects equally quickly—they work at equal effective productivity of 1. Since they are never interrupted, project modularity never comes into play, and since the two watchmakers work at the same speed, they complete their tasks in the same amount of time. At the other extreme, in which the addition of every piece is interrupted, the two watchmakers also work at the same speed—neither is ever able to finish an assembly, since every attempt to add a piece to their assembly is interrupted. Their effective productivity is again equal, now at 0.

To see analytically the implications of the model at intermediate probabilities of interruption, it is first useful to derive the expected time a watchmaker takes to complete a watch. Let  $p$  denote the probability of the watchmaker's being interrupted at each step, and let  $q(p, j)$  be the expected number of parts added to complete a module of  $j$  parts. First, as discussed above, note that if  $p = 0$ ,  $q(p, j) = j$ , since no work is ever lost. Next, if  $p = 1$ ,  $q(p, j) = \infty$ , since no work is ever completed. Finally, for  $0 < p < 1$ ,

$$\begin{aligned}
 q(p, 0) &= 0 \\
 q(p, 1) &= (1 - p) + 2p(1 - p) + 3p^2(1 - p) + \dots \\
 &= (1 - p) \sum_{i=0}^{\infty} (i + 1)p^i \\
 &= \frac{1 - p}{(1 - p)^2} \\
 &= \frac{1}{1 - p}.
 \end{aligned}$$

To generalize this to an arbitrary number of pieces  $j$ , it is helpful to define  $z = q(p, j - 1) + 1$ , one

more than the number of pieces needed to achieve an assembly of  $j - 1$  parts.

$$\begin{aligned}
 q(p, j) &= z(1 - p) + 2zp(1 - p) + 3zp^2(1 - p) + \dots \\
 &= z(1 - p)(1 + 2p + 3p^2 + \dots) \\
 &= z(1 - p) \left( \frac{1}{(1 - p)^2} \right) \\
 &= \frac{z}{1 - p} \\
 &= \frac{q(p, j - 1) + 1}{1 - p}.
 \end{aligned}$$

I can now eliminate the recursion from this expression.

$$\begin{aligned}
 q(p, j) &= \frac{q(p, j - 1)}{1 - p} + \frac{1}{1 - p} \\
 &= \frac{q(p, j - 2)}{(1 - p)^2} + \frac{1}{(1 - p)^2} + \frac{1}{1 - p},
 \end{aligned}$$

and so on until  $q(p, 0) = 0$ , so that

$$\begin{aligned}
 &= \sum_{i=1}^j \frac{1}{(1 - p)^i} \\
 &= \frac{\frac{1}{1 - p} - \frac{1}{(1 - p)^{j+1}}}{1 - \frac{1}{1 - p}} \\
 &= \frac{\frac{1}{(1 - p)^j} - 1}{p} \\
 q(p, j) &= \frac{1}{p(1 - p)^j} - \frac{1}{p}. \tag{1}
 \end{aligned}$$

A more easily interpreted quantity is the effective productivity  $f(p, j)$ , i.e. the number of completed parts per unit time. First,  $f(0, j) = 1$ , since no work is lost. Next,  $f(1, j) = 0$ , since all work is lost. Finally, for  $0 < p < 1$ ,  $f(p, j)$  is given by

$$\begin{aligned}
 f(p, j) &= \frac{j}{q(p, j)} \\
 &= \frac{jp(1 - p)^j}{1 - (1 - p)^j}. \tag{2}
 \end{aligned}$$

$f(p, j)$ , is plotted in Figure 2 as a function of  $p$ . At  $p = 0$ , no interruptions occur, and effective

productivity is always 1, regardless of the modularity of the project design. At  $p = 1$ , the addition of every piece is interrupted, and effective productivity is 0, again irrespective of module size. For intermediate probabilities, the module size has an interesting effect. Note that  $f(p, 1) = 1 - p$ —if each piece is a module, then the effective productivity falls off linearly with the interruption probability. Taking this as a baseline, consider increasing the module size. For  $j > 1$ , the effect of increasing the interruption probability  $p$  is dramatic: effective productivity falls quickly, as is evident in the curve for  $j = 10$  in Figure 2. When modules are bigger, more pieces come disassembled on average each time the worker is interrupted, and the strength of this effect increases quickly with the module size.

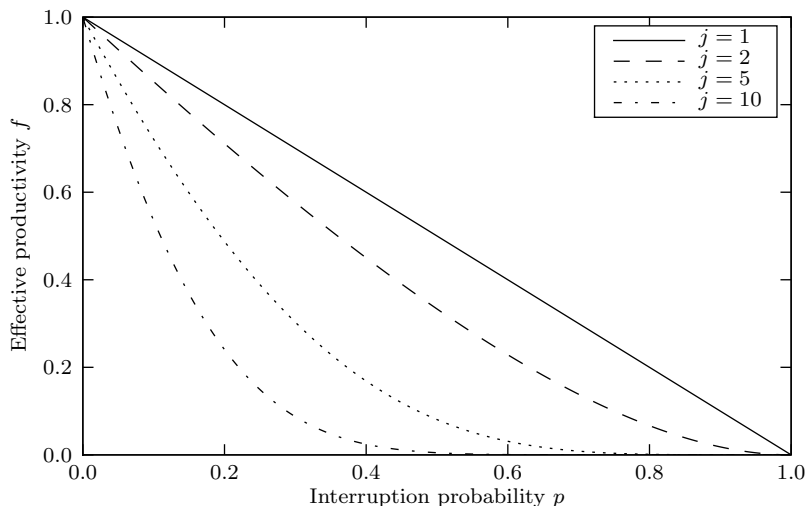


Figure 2: Effective productivity as a function of interruption probability for various module sizes  $j$ . For  $j > 1$ , effective productivity falls quickly as the interruption probability rises.

Figure 2 shows graphically the first implication of the model—that more granular project designs (those with smaller  $j$ ) are much more tolerant of interruptions than are less granular designs. The larger module size of some projects increase the effort wasted at each interruption, lowering overall productivity dramatically even for moderate  $p$ . With small modules, meanwhile, other projects, although they are operating in an identical production environment, waste very little effort at interruptions. As such, these projects can attain a relatively high level of efficiency even at quite severe interruption probabilities. The gain in effective productivity due to the choice of a unit module size

$j = 1$  is shown in Figure 3. Note the gains are the more substantial the larger the alternative module size, and that the maximum increase in module size occurs at lower  $p$  when  $j = 1$  is chosen over higher  $j$ . In other words, large modules pay a high price in efficiency versus very small modules, even in environments of relatively modest interruption.

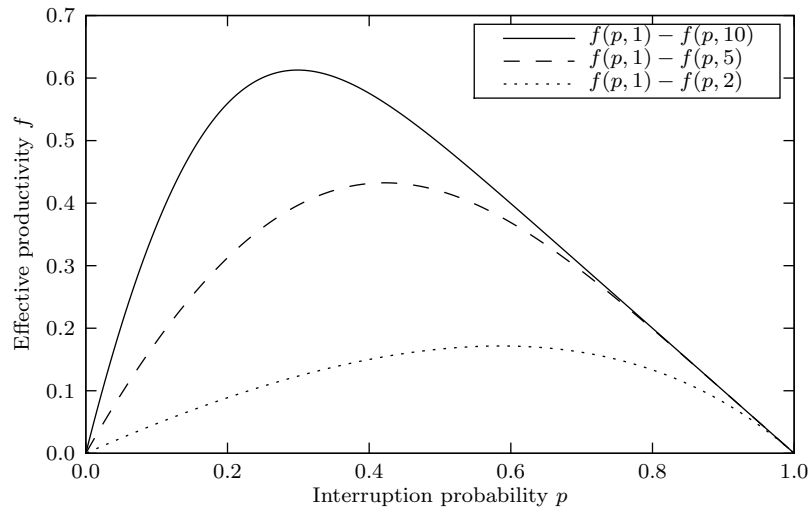


Figure 3: Increase in effective productivity by choosing  $j = 1$  over  $j = 10$  (solid line),  $j = 5$  (dashed line), or  $j = 2$  (dotted line), as a function of interruption probability. For  $j = 10$ , the greatest increase is at  $p = 0.299$ , for  $j = 5$  at  $p = 0.422$ , and for  $j = 2$  at  $p = 0.586$ .

Figure 4 shows the same function  $f(p, j)$ , plotted now as a function of module size  $j$ . Note that the horizontal axis uses a logarithmic scale.

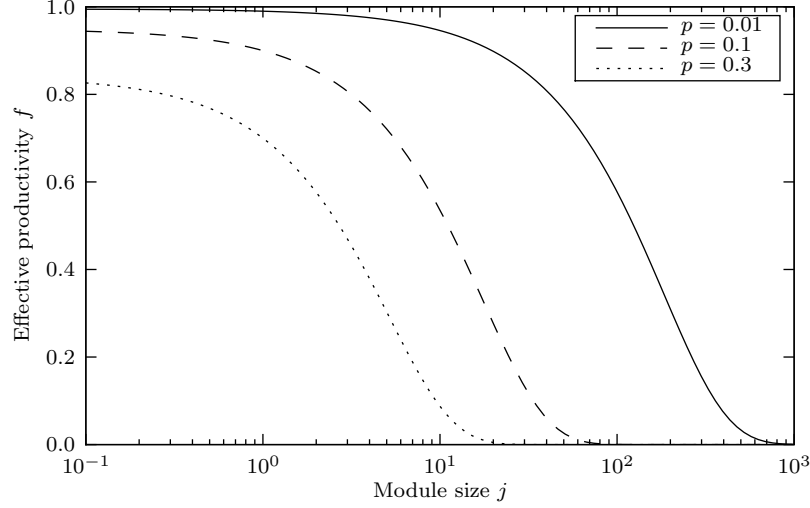


Figure 4: Effective productivity as a function of module size  $j$  for various interruption probabilities  $p$ . At low  $p$ , effective productivity remains high even at very large module sizes. As  $p$  increases, maintaining high effective productivity requires the use of smaller modules. Effective productivity approaches a limit of  $\frac{p}{-\log(1-p)}$  as  $j \rightarrow 0$ .

At very large module sizes, interruption probability makes almost no difference—the waste due to interruption is so great that productivity falls to zero. At intermediate module sizes, effective productivity falls quickly over a small range that varies with the interruption probability. At small module sizes, effective productivity appears to approach a limit at  $j \rightarrow 0$  that varies with interruption probability  $p$ .<sup>15</sup> This is indeed the case, as I now show. I seek to evaluate the limit

$$\lim_{j \rightarrow 0} f(p, j) = \lim_{j \rightarrow 0} \frac{jp(1-p)^j}{1 - (1-p)^j}.$$

<sup>15</sup> For this limit to be meaningful,  $j$  must be allowed to take on non-integral values, yet the derivation above used mathematical induction over integral  $j$  to arrive at the equations used below. However, since no particular definition of the  $j$  tasks making up each module was given, we can suppose that non-integral values of  $j$  are attainable by a suitable scaling of the units by which tasks are counted.

Noting that

$$\begin{aligned}\lim_{j \rightarrow 0} jp(1-p)^j &= 0 \\ \lim_{j \rightarrow 0} 1 - (1-p)^j &= 0,\end{aligned}$$

I employ l'Hôpital's rule.<sup>16</sup> So

$$\begin{aligned}\lim_{j \rightarrow 0} f(p, j) &= \lim_{j \rightarrow 0} \frac{p(1-p)^j + jp \log(1-p)(1-p)^j}{-\log(1-p)(1-p)^j} \\ &= \lim_{j \rightarrow 0} \frac{p(1 + j \log(1-p))}{-\log(1-p)} \\ &= \frac{p}{-\log(1-p)}.\end{aligned}\tag{3}$$

In other words, the interruption probability places an upper bound on effective productivity. Equation (3) is plotted in Figure 5.

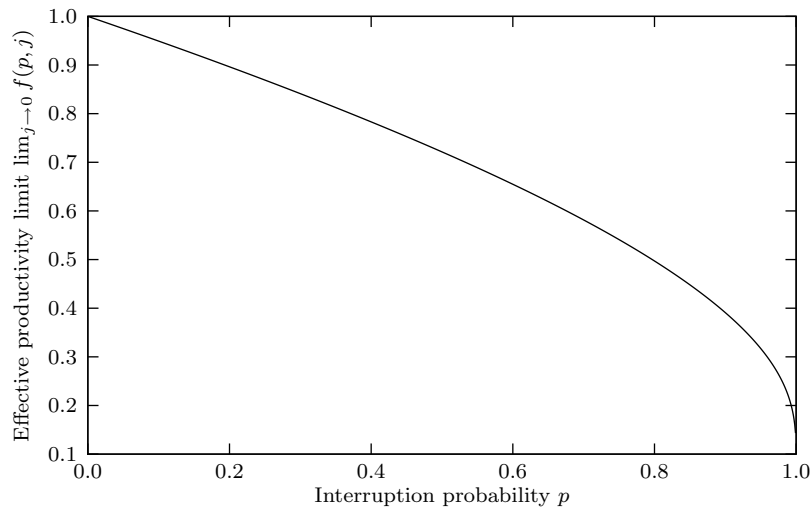


Figure 5: Maximum effective productivity as a function of interruption probability.

An interpretation of these results in the context of peer production may be useful at this point. As

<sup>16</sup>Note that, as required by l'Hôpital's rule, the derivative of the denominator,  $-(\log(1-p))(1-p)^j$ , does not vanish near  $j = 0$ .

---

discussed at the beginning of this section, the model parameter  $p$  is intended to represent frustrating influences on contribution to peer-production projects. Standard firms, where workers are likely to be full-time and managers can exert control, can be thought of as having low  $p$  relative to peer-production projects. The central benefit of a granular (small- $j$ ) project is in its resilience to the effects of high  $p$ . Figure 3 illustrates this—while a firm with very low  $p$  might see little benefit in choosing a smaller module size, an open-source project with moderate  $p$  would see substantial benefit. Lessening the dependencies between contributions greatly increases the capacity to produce in such an environment.

I introduced this model in a discussion of peer production, to which context it is now appropriate to return. Since most open-source and Wikipedia contributors are working on projects in their spare time, the success of the projects depends on their ability to generate meaningful products by working during disjointed intervals of time. To apply the model, suppose that this disjointedness is captured by  $p$ —contributors who are working on open-source projects over lunch or after work have almost by definition less, and less contiguous, time to devote to their projects, or in other words they have higher  $p$ . A successful project relies on capturing what effort they are able to exert during that time, or in terms of the model, on getting as many completed assemblies out in the time that is available; that is, they must maximize effective productivity. Figures 2 and 4, then, offer immediate and concrete lessons.

Figure 4 shows the effect of increasing module size for different interruption probabilities. As modules are made larger, projects with low  $p$  are much better able to maintain high productivity than are projects with high  $p$ . Open-source production, characterized by a high level of  $p$  due to participants' being volunteer, part-time and geographically dispersed, suffers much more from having large module size than does firm-based production, characterized by employees who have low  $p$  by virtue of being paid, full-time, and relatively geographically centralized. This observation has not been lost on open-source project managers.

There is a limit, however, to the gains from modularity, as shown in Figure 5. Faced with a choice between redesigning a project to create smaller modules (i.e. to lower  $j$ , and raising funds to pay workers (i.e. to lower  $p$ ), a manager in the model world should keep this limit in mind. This can be seen in Figure 4. If an open-source project is at a high  $p$ , say  $p = 0.3$ , then redesigning the project (lowering  $j$ , or moving rightward along a curve in Figure 4) can only ever improve efficiency by so much. For some projects, it might make more sense to move to a different curve by finding a

way to lower  $p$ .

Though the model offers insight into modularity in peer-production projects, there is nothing in it that is entirely specific to that context, and some readers may find another context more familiar. Suppose that Tempus and Hora are academic researchers. Tempus, the researcher with the monolithic project design, is a sociologist, required by the standards of his discipline to write 50-page papers. Hora, the researcher with the modular project design, is a physicist, and thus her normal unit of contribution is a letter of perhaps 5 pages. The driving feature of the model, that interruptions cause the loss of the latest incomplete module, should be interpreted loosely—for example, each time the researcher sits down to work on a project, she must devote some time to finding her place in her work materials, and this cost scales with the length of the paper. Tempus’s 50-page sociology paper requires that he exert more effort to keep track of a larger body of writing, more sheets of draft, a more extensive outline, and so on. Paying a cost of this type after each interruption is equivalent to the type of model discussed above.

Given this interpretation, the shape of the curves in Figure 2 has great significance for the productivity of Tempus and Hora as researchers. Supposing an interruption probability of  $p = 0.05$ , the relative productivity of Hora and Tempus is

$$\frac{f(j_{\text{Tempus}})}{f(j_{\text{Hora}})} = \frac{f(0.05, 5)}{f(0.05, 50)} \quad (4)$$

$$= 4.1, \quad (5)$$

meaning that Hora, the physicist, produces more than 4 times as many pages of research per unit time than does Tempus, the sociologist. The figures in this example are chosen simply for illustration, but the broader point is clear: a project’s modularity structure can have a major impact on the productivity of its participants.

### 3.2 The model does not assume a single participant

To relate this model to peer production, it is necessary to revisit an implicit assumption that I have made thus far. The production process described is executed by a single individual, as was the case in Simon’s watchmaker model. Although open-source software can be written by individuals in isolation, *peer production*, which overlaps with but is not identical to open-source, presumes a collaborative process. This was emphasized in the Benkler’s definition given on p. 3 above.

Fortunately, there is a simple multi-agent interpretation of the simple model studied above. In it, modules play the role of stable individual components. When an interruption occurs, only incomplete modules are lost. Suppose that, instead of representing a single individual, Tempus and Hora each represent production teams consisting of  $l$  individuals working on the same project design. I assume that modules are tasks that must be completed by a single individual. This is consistent with the framework laid out above—in particular,  $p$  is intended to represent any forces in the environment frustrating production, including the costs of inter-person coordination.

With this assumption, the adaptation of the earlier results is straightforward. In (1), I showed that a single agent could complete a module of size  $j$  in time

$$q(p, j) = \frac{1}{p(1-p)^j} - \frac{1}{p}.$$

Because there is no interdependence among modules, increasing the number of participant  $l$  simply means that, in expectation,  $l$  modules can be completed in this time rather than just 1. The relevant measure of efficiency in the multi-agent situation is pieces in finished assemblies per participant-hour, which is

$$\frac{lj}{lq(p, j)},$$

which is the same as  $f(p, j)$ . In this simple case, therefore, changing the number of participants has no effect on the quantities of interest. As the model is elaborated below, this will not continue to be the case.

Figure 2 shows the effect of increasing interruption probability for different module sizes. As interruption probability is increased, projects with low  $j$  (the straight line) are better able to maintain high productivity than are projects with high  $j$  (the bottom curve). If the parameter  $p$  is interpreted to include coordination problems stemming from the project's having multiple contributors, Figure 2 says that a project's code is opened to more volunteer contributors, more modular (smaller- $j$ ) projects benefit disproportionately. The more contributors there are, the more difficulty there is in aggregating their contributions. This difficulty manifests itself in the adequate completion of contributors' individual modules, so that the productivity of the organization as a whole benefits from a reduction in the module size.

This discussion has strongly hinted at a limitation of the approach taken thus far. To assume that a production process is modular, but that the modules are not at all interdependent, is to

trivialize the design problem faced by open-source projects. The implication of Figure 2 is that the smallest possible module will be the most productive. Smith referred to a “proper division and combination”, and it is unlikely that the finest possible division will fit the bill. With the multi-agent interpretation given in this section, the model also trivializes the coordination problem faced by the agents. It assumes that no matter how many pieces the project is divided into, their reassembly will not be an impediment to production. These two variations of the same critique of the model are addressed in the next section.

### 3.3 Modularity has costs, too

The model presented above represents the first piece of a framework that I argued captures relevant features of a peer-production system. However, it led to a result that does not sit well—the highest effective productivity is always achieved by using the smallest possible module size. The reason for this result is simple—reducing the module size decreases the amount of work lost to interruptions, but no price is paid for increasing the number of modules. This results above apply to large numbers of tasks that are largely independent. This is not the case, however, in the application of most interest. Intuitively, the smaller the units a peer-production project is divided into, the more work will be required to piece them back together into a coherent whole. In this subsection, I elaborate a simple extension to the model introduced above that incorporates this feature.

In the setup of Section 3.1, suppose that completed modules must themselves be assembled in order to yield a finished product, and suppose that this assembly is itself a process subject to interruption exactly like the assembly of each module. That is, I now consider a hierarchical or two-stage production process, in which a fixed set of components is first assembled into modules, and then the modules are themselves assembled into finished products. This process, as above, takes place in an environment of interruptions. If interruption occurs during the production of a higher-level module, the entire assembly falls apart and each base-level module must be reassembled from component pieces. This specification of the model is strong—it asserts that base-level modules can hold together only long enough to be assembled into a higher-level module.<sup>17</sup>

Assume that the interruption occurs with the same probability  $p$  at each stage of production. Then the time  $Q(p, M, j)$  needed to complete a project of size  $M$  divided into modules of size  $j$  is

<sup>17</sup> Below on p. 36, I consider the alternative assumption, that interruptions cause the loss of only a fraction of the completed work, rather than all of it. There I show that this parameter can safely be disregarded.

given by

$$Q(p, M, j) = q\left(p, \frac{M}{j}\right)q(p, j). \quad (6)$$

The corresponding effective productivity  $F(p, M, j)$ , is given by

$$F(p, M, j) = \frac{M}{Q(p, M, j)}.$$

Effective productivity  $F$  is plotted in Figure 6, which parallels Figure 4 from the previous section.

In Figure 6, I assume a project size  $M = 100$  for the sake of illustration.

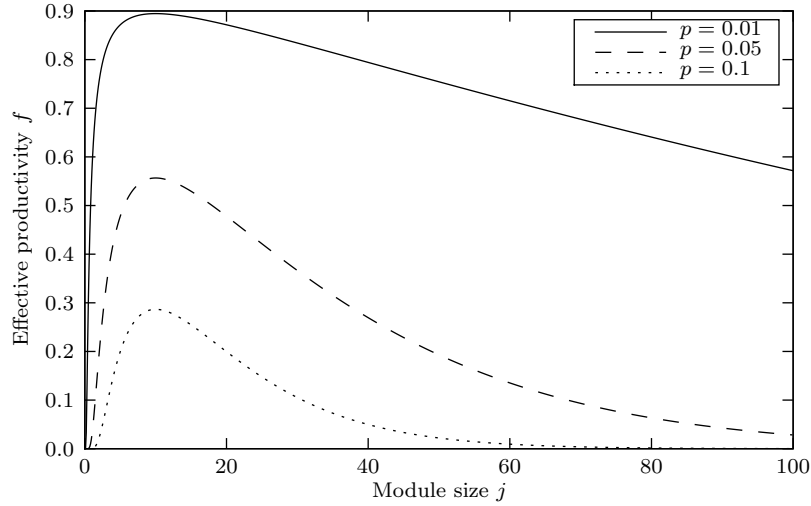


Figure 6: Effective productivity as a function of module size  $j$  for various interruption probabilities  $p$ , two-stage model with  $M = 100$ . Here, productivity falls quickly for very small module sizes, reflecting the added cost of inter-module coordination. Productivity always has a maximum at  $j = \sqrt{M}$ ,  $j = 10$  in this case.

Note that in the two-stage model, effective productivity cannot be arbitrarily increased by decreasing the module size, as was the case with the one-stage model above. Instead, for each interruption probability  $p$ , a maximum effective productivity is achieved at an intermediate module size  $j = \sqrt{M}$ . This result is not difficult to derive from the expression for the completion time  $q(p, j)$  of

a module of size  $j$  given above in (1). There, I had

$$q(p, j) = \frac{1}{p(1-p)^j} - \frac{1}{p}.$$

Using (6) and differentiating,

$$\frac{d}{dj}Q(p, M, j) = \left( \frac{1}{p(1-p)^j} - \frac{1}{p} \right) \frac{1}{p(1-p)^{M/j}} \log(1-p) \frac{M}{j^2} - \left( \frac{1}{p(1-p)^{M/j}} - \frac{1}{p} \right) \frac{1}{p(1-p)^j} \log(1-p).$$

Setting this derivative equal to zero and simplifying,

$$j = \frac{M}{j}, \tag{7}$$

the desired result. Equation (7) can be given an interesting interpretation: the highest-efficiency division of the project into modules is the one that equally divides the task of coordination between the lower level (production of each module) and the upper level (assembly of the modules into a whole). This is essentially a consequence of the homogeneous structure that I have assumed the project to possess—the first stage of production and the second are identical, and are performed by identical workers. The latter assumption can be relaxed to generalize Equation (7). Namely, Figure 7 is similar to Figure 6, but only the top-level interruption probability, denoted  $p_0$ , is varied. The bottom-level interruption probability, denoted  $p_1$ , is held constant at  $p_1 = 0.05$ . The middle curve, then, corresponds to the middle curve of Figure 6.

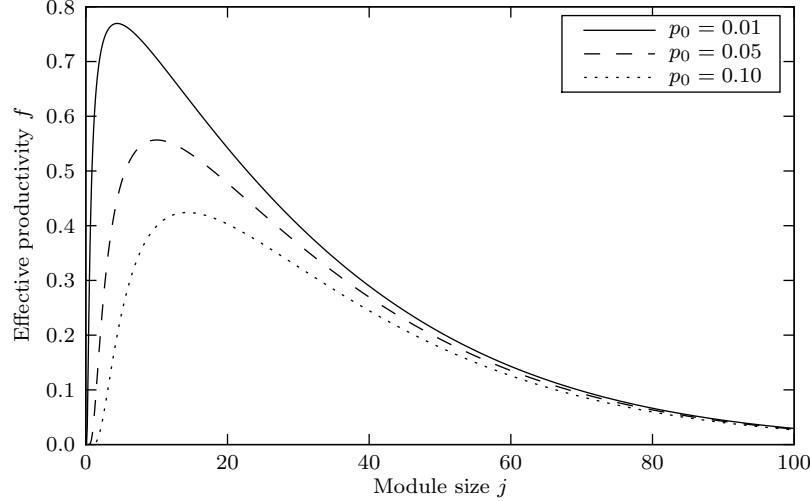


Figure 7: Effective productivity as a function of module size  $j$  for various top-level interruption probabilities  $p_0$ , two-layered model with  $M = 100$ . The optimal module size varies with the second-stage interruption probability  $p_0$ —for  $p_0 = 0.01$ , effective productivity is maximized at  $j = 4.4$ , for  $p_0 = 0.05$  at  $j = 10$ , and for  $p_0 = 0.10$  at  $j = 14.3$ .

Two main effects can be identified in Figure 7. First, increasing the second-stage (top-level) interruption probability has a dramatic consequence for overall productivity—the work completed at each time step falls from about 0.75 to about 0.4 as  $p_0$  rises from 0.01 to 0.1. Second, the optimal module size changes as the ratio between the interruption probabilities of the two levels changes. When  $p_0 = p_1 = 0.05$ , maximum effective productivity is attained at  $j = \sqrt{M} = 10$ , as shown above. When  $p_0 < p_1$ , the optimal  $j$  is less than this, and *vice versa*. For example,  $p_0 = 0.01 < p_1$  has an optimal  $j = 4.4$ , while  $p_0 = 0.10 > p_1$  has optimal  $j = 14.3$ .

This relationship is elaborated in Figure 8. Each curve corresponds to a given first-stage interruption probability  $p_1$ . On the horizontal axis, I show the second-stage interruption probability  $p_0$ . On the vertical axis, I show the optimal module size for the corresponding value of  $p_0$ . Note that, as implied by Equation (7), whenever  $p_0 = p_1$ , the optimal  $j$  is 10 (the horizontal line). Consistent with the discussion above, as  $p_0$  is decreased from this value, the optimal module size decreases, shifting the burden toward the top level. As  $p_0$  is increased, the optimal module size increases, shifting the burden toward the bottom level.

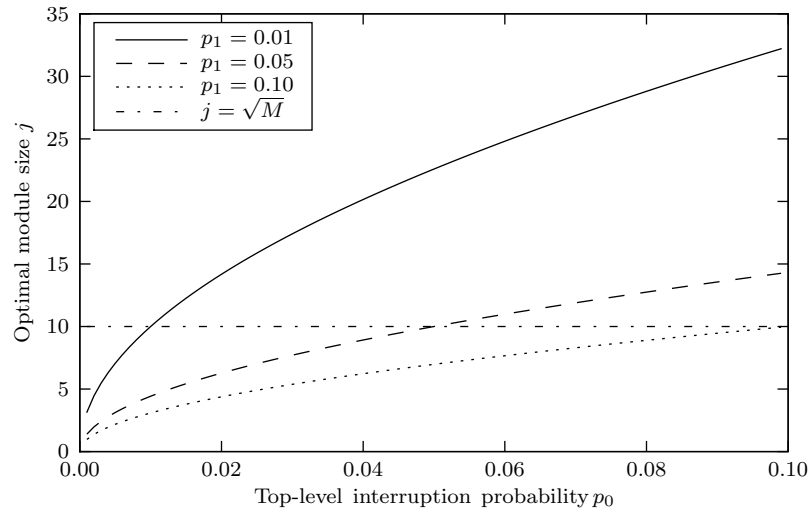


Figure 8: Optimal module size  $j$  as a function of top-level interruption probability  $p_0$ , various bottom-level interruption probabilities  $p_1$ , two-layered model with  $M = 100$ .

Observing the similarity in shape between the three curves in Figure 8, a natural question is whether the Figure 9 shows the same curves as does Figure 8, but plotted as a function of the ratio of interruption probabilities  $\frac{p_0}{p_1}$ .

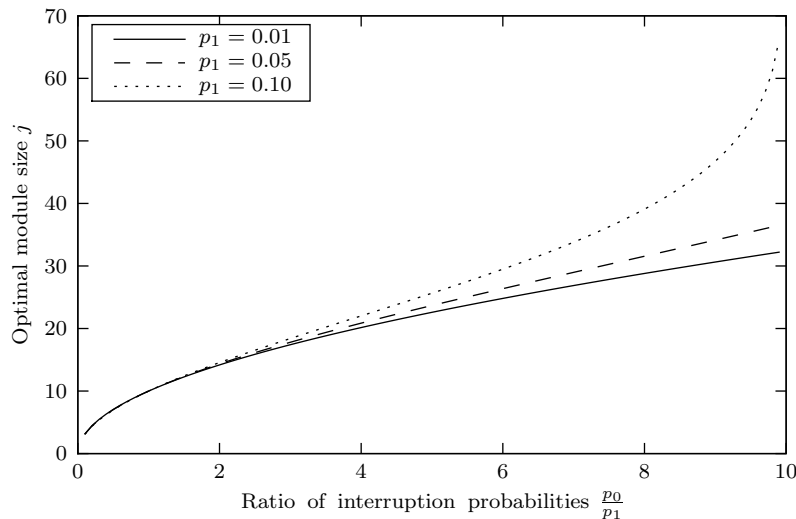


Figure 9: Optimal module size  $j$  as a function of ratio of second- to first-stage interruption probabilities  $\frac{p_0}{p_1}$ , various bottom-level interruption probabilities  $p_1$ , two-stage model with  $M = 100$ .

I conclude this section with some interpretation. In the preceding section, I showed that decreasing a project’s module size increased its resilience to interruptions. A peer-production project, that is, should be broken down as finely as possible. Taken to the extreme, however, that recommendation becomes implausible, since it ignores the cost of reassembling the modules. In this section, I used the logic of the model to address this by taking into account the difficulty of assembling the project from its components. When this is incorporated, the best choice of module is intermediate. Specifically, when the interruption probability is the same at the first and second stages of production—when the task of creating a project out of its components faces the same level of challenges as does creating the components themselves—the best choice of module size is the one that divides the work equally between the first and second stages of production. Often the functions of a piece of software can be implemented either at a basic level, written from first principles, or at a high level, assembled from other functions. The model implies that, when given a choice, the preference should be for an equal division of effort between those two tasks, placing too great a burden on neither the base level nor the synthetic level.

When a different production environment is faced at the two stages of production, adaptation

should be made to the division of labor between the first and second stages of production. Specifically, when the second stage can be performed with lower  $p$ , that is to say with fewer influences frustrating production,  $j$  should be lowered—modules should be shrunk—to shift the burden to that stage. If, by contrast, the first stage had the lower  $p$ ,  $j$  should be raised to shift the burden of production in the other direction.

The reCAPTCHA example from p. 9 above illustrates this point nicely. The overall production effort is described quite naturally by this model—the first stage of production is the recognition of individual words, while the second stage is their synthesis into entire texts, including confirming the recognition of the words, spell-checking, and so on. Individual users are asked to perform the first stage of this production in the course of their routine tasks online, by recognizing and typing in the word they are presented with. These users face a high  $p$ , since they are probably not willing to offer more than a second’s worth of effort before giving up and going to another website. The second stage of production is performed by dedicated servers and individuals operating them, or in other words in an environment of low  $p$ . Accordingly, the division of labor between the two stages of production is wisely allocated so that the lower- $p$  participants bear the greater burden in the production process. That is, the module size of the individual contribution is very small— $j$  is very low. As such, the number of these contributions is high, but the task of dealing with these contributions is given to the project participants who face the lower likelihood of being interrupted.

As before, an interpretation of this model in the context of academic research may also be instructive. Using the same parameters as above, consider a sociologist who writes papers 50 pages in length and a physicist who writes papers five pages in length. The added level of coordination in this version of the model can be thought of as the effort needed on the part of the physicist to synthesize several letter-length publications into a more comprehensive journal article. Again comparing effective productivities, now using Equation (3.3),

$$\frac{F(j_{\text{Tempus}})}{F(j_{\text{Hora}})} = \frac{F(0.05, 50, 5)}{F(0.05, 50, 5)} \quad (8)$$

$$= 3.2. \quad (9)$$

In this version of the model, the physicist Hora still achieves a higher productivity than the sociologist Tempus. Both produce 50 pages of output, but Hora’s is structured into five-page projects, meaning that less work is put in jeopardy by the intrusion of outside distractions. As a result, Hora produces

more than three times as much output for each unit of time. Note that the comparison is of effective productivities. I have measured this as number of pages per unit of time, supposing therefore that a page represents a comparable amount of output regardless of the subject matter. Thus the physicist's advantage in productivity is in her *output in pages*, not in the number of papers she writes. The numerical example above could equivalently state that in the same amount of time in which the sociologist Tempus produces 50 pages of output, the physicist Hora produces 160 pages.

## 4 Hierarchical production, multiple agents: simulation model

The hierarchical model of modular production discussed in the preceding section addresses one limitation of the non-hierarchical model; namely, it incorporates the coordination cost of excessive subdivision of a project into modules. As the modules become too small, it becomes harder to assemble them into a working product. To draw conclusions from the model for peer production, one further limitation needs to be addressed—the organization of potentially large numbers of individuals into a coherent productive team. The model discussed above implicitly limited the number of project participants through its assumption on how modules are assembled at the second layer of the hierarchy. In reality, however, peer-production projects may have more or fewer participants available to them, and these participants may be organized in a variety of ways, and the openness of this participation presents one of the most intriguing features of the phenomenon. In this section and the section that follows, simulation is used to adapt the mechanisms of the preceding model to allow the study of the effects of project participation. First, in this section, I introduce the simulated version of the model and show that it reduces to the analytical model for appropriate parameter choices. I then consider some parameters that are omitted from the model, and argue that their omission does not fundamentally change its behavior.

### 4.1 Details of the simulation model

The simulation implements a version of Simon's watchmaker model that employs a flexible number of individuals. The project consists of a number of components  $M$  and a number of participants  $N$ . The participants are organized into a hierarchy with branching ratio  $b$ , which is to say that each member of a level other than the lowest has  $b$  subordinates at the next lower level. At the bottom level, individuals assemble components into modules of size  $j$ , each capable of adding components

to their assembly at the same rate. At each time step, each participant is randomly interrupted with probability  $p$ , with interruptions occurring independently across participants. If interrupted, the participant loses the module on which he or she is currently working.

At the same time, a single agent at the next stage in the production process is working on a higher-order assembly. Whenever an agent at the base level finishes an assembly, its ‘parent’ or superordinate in the hierarchy is able to add it to its own higher-order assembly. This agent is also subject to interruption, at some probability, possibly the same  $p$  as at the base level. If the second-order agent is interrupted, any incomplete second-order assembly is lost completely, as are all the first-order assemblies that make it up.

The configuration of project participants described above is illustrated in Figure 10. There,  $N = 21$  participants are arranged into a hierarchy with branching ratio  $b = 4$ . A lone individual at the top level has four subordinates, each of whom in turn has four subordinates. A difficulty is sometimes encountered when a particular branching ratio  $b$  cannot be achieved for a given  $N$ , specifically when

$$\sum_{k=0}^K b^k$$

does not equal  $N$  for any  $K$ . In such situations, I have elected to distribute participants so that

- all participants at all levels other than the bottom have at least one subordinate, and
- no participant at any level has more than one fewer subordinate than any other participant at the same level.

This is illustrated in Figure 11. Here  $N = 19$  participants are allocated into a hierarchy, again with branching ratio  $b = 4$ . In the middle layer, two participants have only 3 subordinates. Thus  $N = 19$  exactly, and the average branching ratio achieved in this tree is somewhat less than 4. For some parameter choices, there is no hierarchy which satisfies these two requirements. In this paper, such choices will not appear.

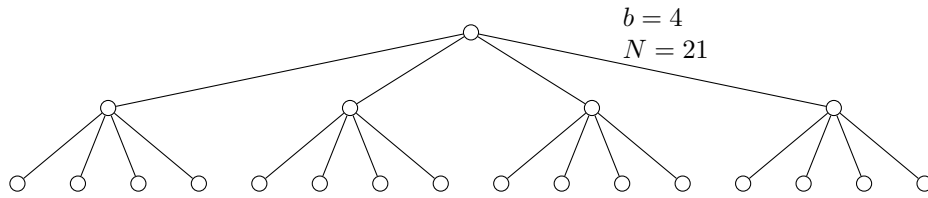


Figure 10: Arrangement of multiple agents into a hierarchy. In this example,  $N = 21$  agents are arranged into a hierarchy with branching ratio  $b = 4$ .

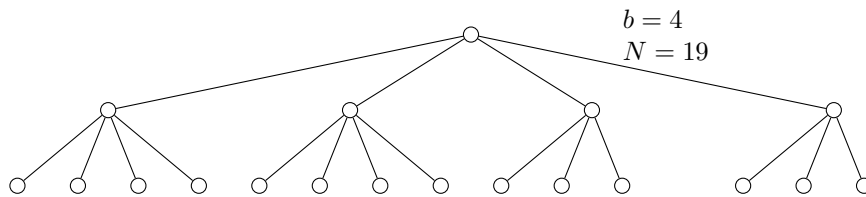


Figure 11: Arrangement of multiple agents into an unbalanced hierarchy. In this example,  $N = 19$  agents are arranged into a hierarchy with branching ratio  $b = 4$ .

The hierarchical production model considered in this paper may call to mind a model studied by Radner (1993). The principal distinction between that model and this one is the modular task structure employed here. Where Radner focuses on associative tasks such as addition, the type of project studied in this paper has an explicit modular structure that can be thought of as externally given. Nonetheless, my model is very much in the same spirit as that of Radner.

## 4.2 Comparison to the analytical model

Intuitively, this simulation implements the same algorithm studied analytically above. To verify that the two approaches are indeed comparable, Figure 12 shows effective productivity  $f$  as computed analytically (solid line) and as calculated in the simulation (dots), for  $p = 0.05$  and  $M = 100$ .  $b = 1$  and  $N = 2$  in the simulation, in order to correspond as closely as possible with the analytical approach. From the figure, it is clear that the two processes are indeed the same at the plotted points.

In Figure 12, effective productivity for the simulated version is not plotted for every value of

$1 \leq j \leq 100$ . The simulation continues until  $M$  components are attached to the assembly. When the module size is not an integral divisor of the project size, some components are wasted. For example, when  $M = 100$  and  $j = 11$ , nine modules would be insufficient to complete the project, but ten modules leads to a waste of ten components. This integer effect is not present in the analytical version, which implicitly allows fractional modules. If all values of  $j$  were plotted for the simulated model in Figure 12, the line would be seen to deviate below the analytical version as  $j$  increased, reverting to the derived function at  $j = \lceil \frac{M}{i} \rceil$ ,  $i \in \{1, 2, \dots\}$ . This figure, then, shows only those values of  $j$  most comparable to the analytical version.

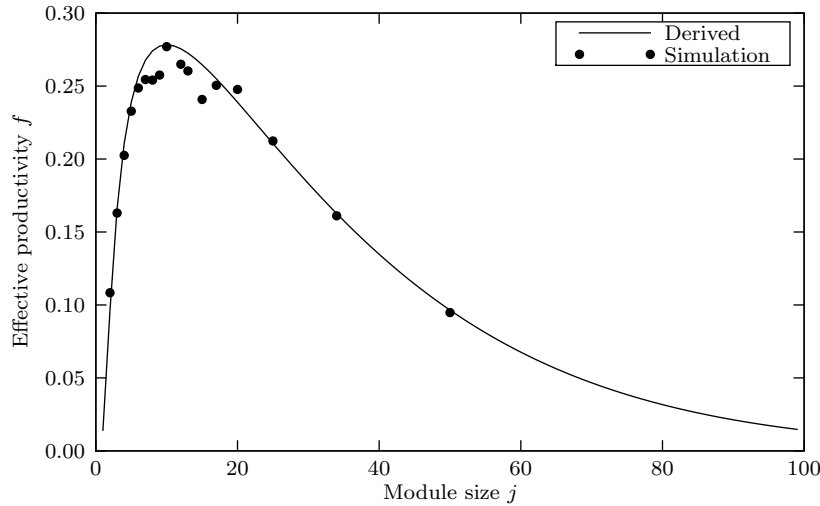


Figure 12: Effective productivity as a function of module size  $j$  in the analytical and simulated two-stage production models. In both versions, interruption probability  $p = 0.05$  and project size  $M = 100$ . In the simulated model,  $N = 2$  and  $b = 1$ , corresponding to a single individual at each stage of production, for purposes of comparison. The simulated version is plotted only at a subset of values of  $j$ , namely  $j = \lceil \frac{M}{i} \rceil$ ,  $i \in \{1, 2, \dots\}$ . For the analytical version, effective productivity has been expressed as an output per individual, an adjustment that was not made in Section 3.3.

Figure 13 shows effective productivity  $f$  as a function of module size  $j$ . In these simulations, I suppose that there are  $N = 2$  participants as in Figure 12. A different value of the top-level interruption probability  $p_0$  is used for each curve, while the second-level interruption probability is held fixed at 0.05. The center curve, corresponding to  $p_0 = p = 0.05$ , thus corresponds to Figure 12.

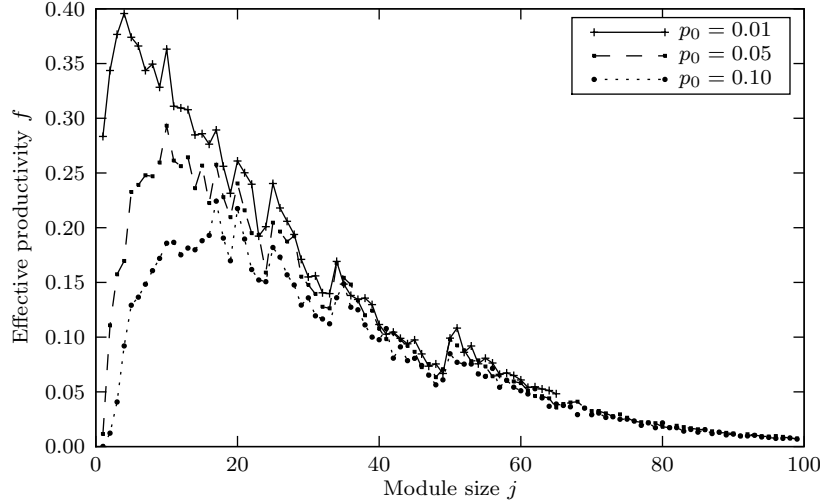


Figure 13: Effective productivity as a function of module size  $j$  in the simulated two-stage production model for various interruption probabilities at the top level  $p_0$ . For each simulation, interruption probability  $p = 0.05$  at the bottom level, project size  $M = 100$ , and the number of participants  $N = 2$ .

Note that lowering  $p_0$  from 0.05 to 0.01 has two effects. The first, a significant overall increase in productivity, is to be expected—the top-level individual plays a central role in the production process, and interruptions at that level will create bottlenecks throughout the hierarchy. The second effect is more subtle. With  $p_0 = 0.01$ , the maximum effective productivity occurs at a much smaller module size,  $j = 4$ , than with  $p_0 = p = 0.05$ , where the maximum is at  $j = 10$ . This figure thus extends the results of Section 3.3. There I showed that, for  $p_0 = p = 0.05$ ,  $j = \sqrt{N} = 10$  achieves a maximum effective productivity by most efficiently sharing the workload between the first and second stages of production. Here, I show that the ideal strategy for sharing that workload depends on the interruption probabilities of the individuals. The module size should be chosen so that the relatively more interrupted individual bears less of responsibility for completing his or her stage of the production process. The curve for  $p_0 = 0.10$  confirms these results. Compared to  $p_0 = 0.05$ , productivity is everywhere lower, and the maximum productivity is higher, at  $j = 17$ , which puts more of the workload on the less-interrupted lower level of the hierarchy. This result further confirms the correspondence between the analytical and simulation models, corresponding

as it does to Figure 7 above.

Figure 14 extends the results of Figure 13, comparing the consequences for effective productivity of lowering the interruption probability at the first and second stages of the two-stage production process. Note that, for small module sizes, efficiency is better served by reducing the interruption probability at the second stage, but for large module sizes, it is preferable to reduce the interruption probability at the first stage. This follows from the structure of the model—since the overall project size  $M$  is fixed, increasing the module size, *ceteris paribus*, increases the burden on the first stage of production. Thus for large  $j$ , there is more benefit to decreasing the interruption probability of the first stage,  $p_1$ . For small  $j$ , by contrast, more burden is placed on the second stage, and so the greater benefit comes from decreasing the interruption probability of the second stage,  $p_0$ .

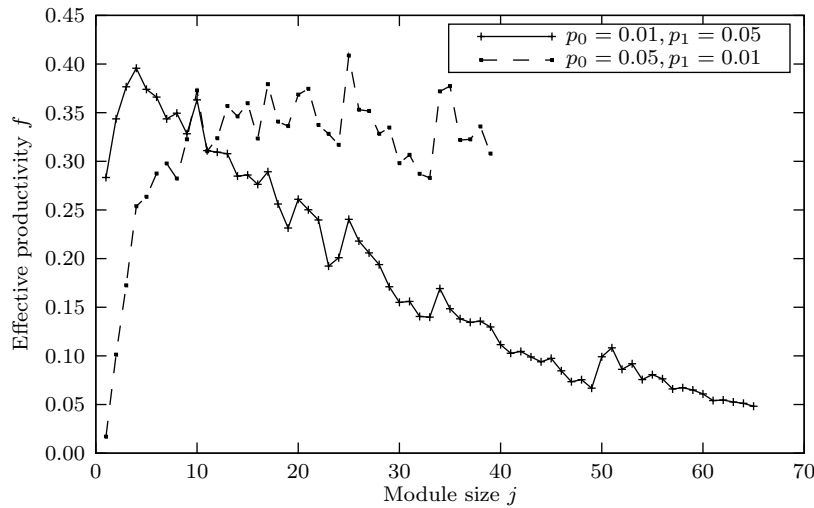


Figure 14: Effective productivity as a function of module size  $j$  in the simulated two-stage production model, with lowered  $p$  at the first (dashed line) and second (solid) level. For the solid (resp. dashed) line, interruption probability  $p = 0.01$  at the second (first) level and  $p = 0.05$  at the first (second) level. Project size  $M = 100$ , and the number of participants  $N = 2$ .

### 4.3 Anticipating some objections to the model

One feature of the simulation model that may seem troubling at first glance is the fact that, upon interruption, participants lose the entire module on which they were working. This assumption

---

seems strong—not all projects are finely-tuned watches, where constant attention is required to keep half-complete modules from falling apart. A mere telephone call or e-mail is not enough to completely wipe out an academic paper or software project. To determine whether this assumption is a grave limitation, I considered a model parameter  $r$ , which represents the fraction of the module a participant retains after interruption. In Simon’s original model and in the version discussed above,  $r = 0$ . If, as  $r$  varies from 0 to 1, qualitative differences in the behavior of the model obtain, then  $r$  must be carefully considered in the versions of the model that follow. The remainder of this section seeks to alleviate this concern. To some extent the argument anticipates results and discussion to follow in Section 5. However, it seems appropriate to address this objection at this stage of the exposition.

Figure 15 shows the first step toward addressing this issue. Using the simulated model, I allow  $r$  to vary from 0 to 1, that is, from the extreme of losing all work at each interruption to the extreme of losing no work at each interruption. As  $r$  goes to 1, note that effective productivity loses its dependence on  $j$ . Recall from the discussion of the analytical model in Section 3 that the productivity loss due to a project’s having a large module size comes from the large amount of work that is lost at each interruption. As  $r$ , the fraction of completed work that is not lost after an interruption, is increased, this effect of having a large module size is diminished—the cost of an interruption declines.

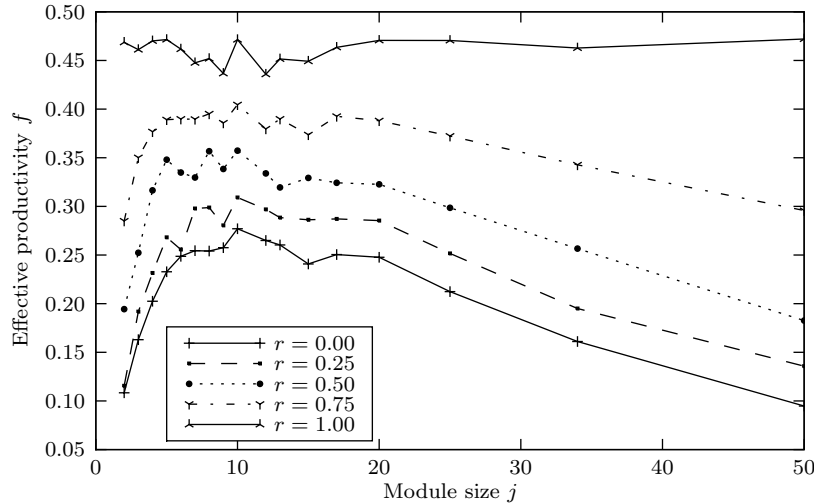


Figure 15: Effective productivity  $f$  as a function of module size  $j$  in the simulated two-stage production model. In this simulation, interruption probability  $p = 0.05$  and project size  $M = 100$ .  $r$ , the fraction of completed work that participants retain after each interruption, varies from zero to one. Increasing  $r$  eliminates the dependence of  $f$  on  $j$ .

One further concern about the parameter  $r$  should also be addressed. Even though variation in  $r$  does not pose any problems through interaction with  $j$ , it could be the case that, when the branching ratio  $b$  of the hierarchy of project participants is allowed to vary, different choices of  $r$  make a qualitative difference. This concern is addressed by Figure 16, which shows effective productivity  $f$  as a function of branching ratio  $b$ , for various values of  $r$ . Observe that, as  $r$  is varied from 0 to 1, no qualitative difference in the dependence of  $f$  on  $b$  is evident. Certainly allowing more work to be retained after each interruption improves productivity, but the effect of variation in this parameter does not appear to depend on  $b$ . The dramatic downward spike in Figure 16 is the consequence of an imbalanced hierarchy that occurs with this particular parameter configuration, and does not bear on the importance of the parameter  $r$ .

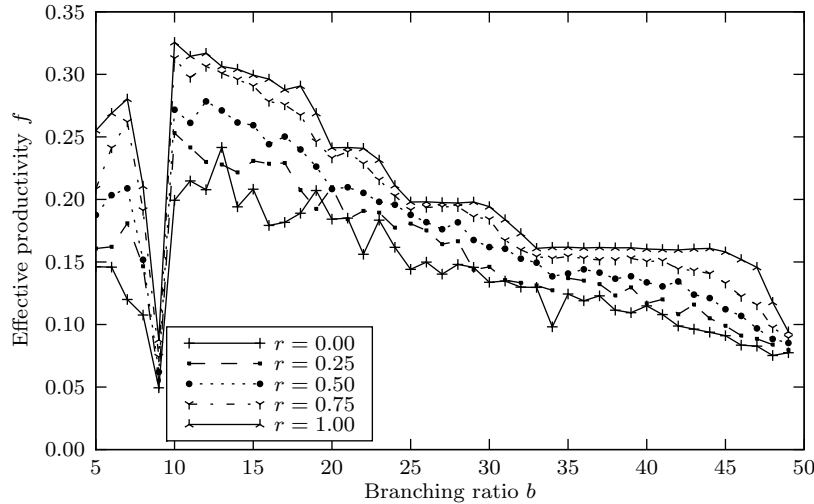


Figure 16: Effective productivity  $f$  as a function of branching ratio  $b$  in the simulated multi-stage production model. In this simulation, interruption probability  $p = 0.05$  and project size  $M = 100$ .  $r$ , the fraction of completed work that participants retain after each interruption, varies from zero to one. Increasing  $r$  increases productivity, but does not affect the dependence of  $f$  on  $b$ . The downward spike in all five series at  $b = 9$  is the consequence of a particularly inefficient imbalanced hierarchy that occurs at that parameter configuration. It does not bear on the parameter  $r$ .

In light of Figures 15 and 16 and the discussion in this section, I disregard the parameter  $r$  for the remainder of this paper.

A final potential objection relates to the operation of the interruption process, which drives most of the results presented here. When, in the hierarchical model, a project participant is interrupted, what should the effect be on his subordinates? There are two possibilities—either they, too, are interrupted, each subordinate of the interrupted agent starting its current module again from scratch; or only the interrupted agent must restart his work, and its subordinates continue uninterrupted. It is not obvious to me which of these alternatives is preferable. Supposing that interruptions propagate downwards through the hierarchy seems to run counter to the idea of modularity that motivates the theoretical approach taken here. A modular production process is based on the idea that interruptions to one component do not affect other components; that individuals manage their collaboration by designing their project so as not to interfere with one another.

At the same time, supposing that an agent's subordinates can continue their work uninterrupted

---

when the agent is interrupted seems to run counter to the idea of hierarchical production. If an agent's interruptions do not affect its subordinates' work, the hierarchical dependence of the pieces of the project is strongly reduced—in this situation, the project is much more *sequential* than it is *hierarchical*. Specifically, if interruptions do not propagate down the hierarchy, problems later in the production process never affect the work done earlier in the process. Any module that is completed is only ever affected by interruptions that come later in the process.

The resolution to this issue is not readily apparent.

## 5 The shape of the hierarchy matters

The simulation model just described is based around five parameters—project size  $M$ , module size  $j$ , number of participants  $N$ , branching ratio  $b$ , and interruption probability  $p$ . For the purposes of this investigation, I employ only  $M = 100$  and  $M = 1000$ , choosing to focus instead on the effects of variation in other parameters while holding project size constant. Though  $p$  could vary widely, I limit consideration to two values,  $p = 0.01$  and  $p = 0.05$ . The absolute choices of  $p$  are not significant, due to the abstract nature of the model. Rather, of interest is the change in the behavior of the system as  $p$  moves from a “low” value to a “high” value.

The other three parameters, module size  $j$ , number of participants  $N$ , and branching ratio  $b$ , are more likely to fall under the direct control of designers of peer-production systems. Given that a modular project design can dramatically increase production efficiency, the more so the more factors are inhibiting production, how can a production team consisting of several individuals be organized so as to make the most of their work? In this section, I use the simulation model to suggest an answer to this question, namely, that for two significant configurations of the parameters  $N$  and  $b$ , I show that deep hierarchy (low  $b$ ) is preferable for small module size, while a shallow hierarchy (high  $b$ ) is preferable for large module sizes. This result, and its interpretation, are discussed below and in Section 6.

A natural question that can be addressed using the simulation model, and these additional parameters it allows to be added, is whether the structure of the peer-production organization changes qualitatively its productive efficiency. In fact it does, as the following results illustrate. Here I consider two sets of simulations which differ only in the organization of participants. Specifically, I hold constant the number of participants  $N$ , along with all other model parameters, while varying

$b$  between the extremes of a very wide and a very deep hierarchy of participants. The first set of simulations uses  $N = 7$ , chosen for a reason that will become clear shortly. The two extreme organizational structures studied are parametrized by  $b = 2$  and  $b = 6$ , as shown in Figure 17. The figure also shows the logic behind the choice of  $N = 7$ ; it is the largest  $N$  such that  $N$  participants can be arranged into a hierarchy with  $b = 2$  in three levels. Below I show that similar results hold with  $N = 15$ , the corresponding choice for four levels.

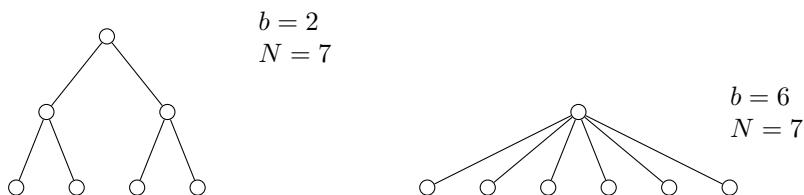


Figure 17: Two arrangements of  $N = 7$  agents into hierarchies. On left,  $b = 2$  corresponds to the deepest possible arrangement. On right,  $b = 6$  corresponds to the broadest possible.

Figure 18 shows the results of simulation of the model on these two hierarchies for a variety of values of module size  $j$ . Disregarding the localized spikes at single values of  $j$ , this figure paints an interesting picture. At small module sizes, the deep hierarchy, corresponding to the solid line, produces much more efficiently. This efficiency drops nearly monotonically as  $j$  increases. The broad hierarchy, represented by the dashed line, varies quite differently in  $j$ . Such a hierarchy is nearly paralyzed by small module sizes, but productivity rises quickly with increasing  $j$ . Interestingly, for small module sizes, the deep hierarchy is preferable, by a large margin. For large module sizes, the broad hierarchy is the better choice. Regardless of the choice of  $b$ , productivity falls off at extremely large module sizes, consistent with results from previous sections.

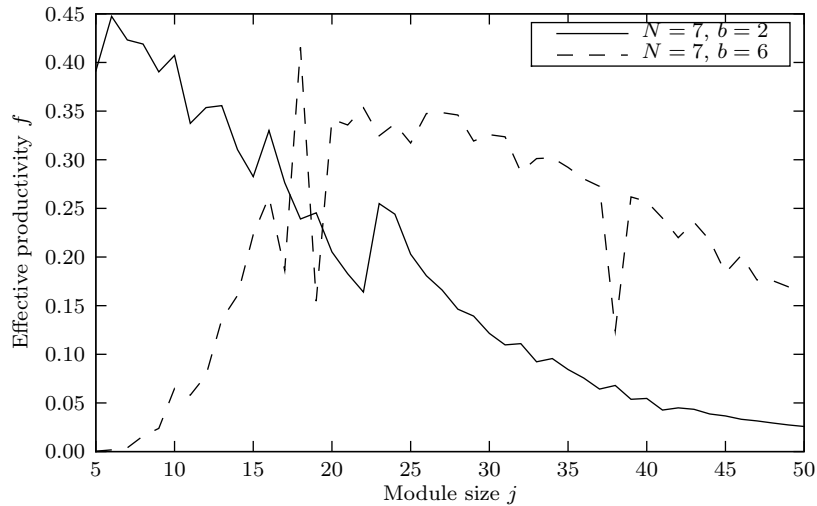


Figure 18: Effective productivity  $f$  as a function of module size  $j$ . The solid line corresponds to  $N = 7$ ,  $b = 2$ —a deep hierarchy. The dashed line corresponds to  $N = 7$ ,  $b = 6$ —the same number of individuals organized into a shallow hierarchy. For both simulations,  $M = 1000$  and  $p = 0.05$ . Subordinates are not interrupted when their parents are interrupted.

Figure 19 repeats this analysis for  $N = 15$ , again using the two extreme branching ratios,  $b = 2$  and  $b = 14$ . Note that, for the larger number of participants used here, a smaller interruption probability  $p = 0.01$ , was used. The simulations can be quite time-consuming with large values of  $p$ . Qualitatively, the dependence of effective productivity for the two sets of simulations is the same as for  $N = 7$  above. For small module sizes, the deep hierarchy is preferable, while for large module sizes, the broad, shallow hierarchy is more efficient.

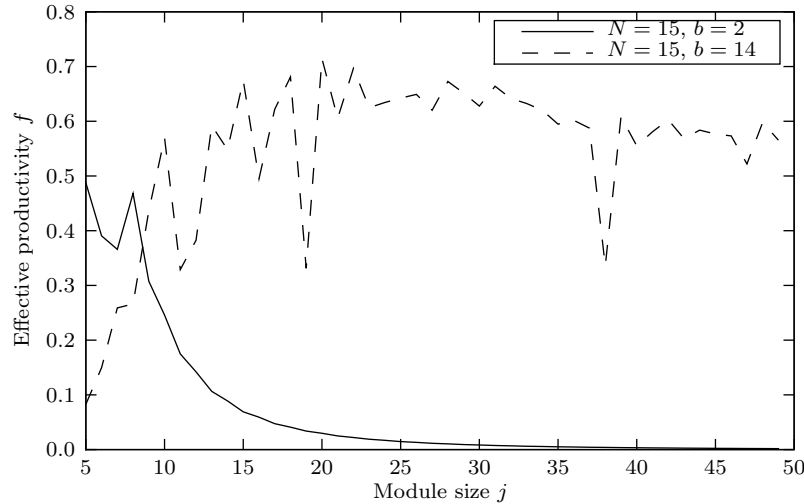


Figure 19: As in Figure 18, but with  $N = 15$ . Effective productivity  $f$  as a function of module size  $j$ . The solid line corresponds to  $N = 15, b = 2$ . The dashed line corresponds to  $N = 15, b = 14$ . For both simulations,  $M = 1000$  and  $p = 0.05$ . Subordinates are not interrupted when their parents are interrupted.

These results have implications for designers of peer-production organizations. The precise lesson to be learned depends on which of the model parameters are under the control of the designer. For example, an open-source project can operate over long periods of time with a relatively fixed code base, while individual participants may join and leave the project during that time. The relevant decision for such a project, then, is not what  $j$  to choose (as  $j$  is determined by the relatively fixed design of the code), but rather where in the hierarchy to incorporate new participants as they join. This corresponds to a choice of which curve to be on in Figures 18 and 19. The results presented here suggest that, for projects with the smallest of module sizes, new participants should be used to deepen the hierarchy. For projects with large module sizes, by contrast, the performance of the deep hierarchy quickly falls off, so that new participants should be used to broaden the hierarchy for any large  $j$ .

Under other circumstances, the choice might be not how to structure participants, but rather how to design the project. For example, in a situation where the organization of participants is fixed by external factors, these simulations show that the ideal choice of module size depends on the depth

or breadth of the project hierarchy. This corresponds to choosing where on the horizontal axis to be, given one of the curves in Figures 18 and 19. The simulations show that, for deep hierarchies (low  $b$ ), efficiency is dramatically improved by selecting a small  $j$ . For shallow hierarchies, by contrast, larger  $j$  can be maintained efficiently. Indeed, these high- $b$  hierarchies are much more resilient to increases in the module size, with efficiency falling off much more slowly with increasing  $j$ .

## 6 Conclusions

### 6.1 Lessons for designers of open-source systems

Open-source software has gained in credibility with the successes of notable projects such as Firefox, Linux, and Apache. Related projects, especially Wikipedia, have produced (perhaps surprisingly) high-quality output from volunteer participation. Social-networking or so-called Web 2.0 websites such as Facebook, Digg, and Flickr provide evidently valuable services (staying in touch with friends, sharing interesting content on the web, and organizing and sharing photos, respectively) by cleverly aggregating users' voluntary contributions. It is not surprising that this model has attracted attention, as these projects appear get something from nothing—in each case, at least one principal input is provided for free. Thus, the CIA has expressed interest in developing an internal wiki (Thompson, 2006), Sun has open-sourced its Java programming language<sup>18</sup>, and media websites now offer tools to allow their content to be shared on social-networking websites.

The model studied in this paper is quite simple and makes no assumptions about the results of the process being modeled; real peer production efforts are complex, varied, and culturally dependent. Moreover, they are path-dependent and subject to trends and fads, as evidenced by the triumph of Myspace over Friendster, and of Facebook over Myspace. Indeed, the entire phenomenon is quite young, with the age of open-source software bounded above by that of the internet, switched on in 1969 (as ARPANET). Thus it seems wise to be modest in attempting to generalize. Within the bounds of the simple model of hierarchical modular production studied here, several conclusions can nonetheless be drawn which may be of relevance to those who wish to create peer-production projects, or to manage or alter existing projects.

The key implication of the one-person analytical model of Section 3.1 is that modularizing production increases efficiency. Dividing a large task into mutually independent subtasks allows the

---

<sup>18</sup> <http://www.sun.com/2006-1113/feature/>, retrieved on 14 January 2009.

---

worker to proceed efficiently, even amidst interruptions. This is a lesson well learned by the software industry, which values having many small, functionally coherent units of code over a single monolithic procedure. The same preference is encoded into the Unix ethos, where the any given tool is expected to do a single job, do it well, and to interface easily with other tools. If hackers have taken this modularity to heart, however, others have yet to do so. In academia, social scientists are quite familiar with the burden of producing large papers as the standard unit of academic production. The example at the end of Section 3.1, though a simplification to say the least, shows that reducing the number of pages in the unit of contribution can dramatically increase the number of pages a researcher is able to write. Smaller units of contribution allow greater, sometimes much greater, productivity.

This version of the model is simplistic, however, in that it assumes that projects can be freely subdivided. In fact, if the aims of the project are fixed, then subdivision comes at the expense of ease of reassembly. The two-stage production model of Section 3.3 shows that when this is taken into account, the appropriate response is to strike a balance, that is, to choose a module size that balances the benefits of small modules with the cost of their reassembly. In the model this balance can be quantified, but in reality it is more likely to serve as a rule of thumb. In a natural way, the right balance to be struck between large and small modules depends on participants' ability to attend to their tasks. If the later stages of production, the equivalent of managers, have the relatively more fragmented time, the balance should be shifted away from them. If, by contrast, the earlier stages have the relatively more fragmented time to contribute to the project, the burden should be shifted away from them.

This version of the model captures, in a sense, the notion of reuse that is central to open-source software and to many other peer-production efforts. By conceiving of their projects as a set of relatively independent tasks, programmers are also providing useful bits of code that can easily be reused, possibly in contexts quite different from their original purpose. The homogeneity of the two-stage model, where the top-level participant assembles the output of the bottom level just as the bottom-level participant assembles atomic components, naturally incorporates this approach to production.

From the simulation results of Section 5, the central lesson is one that was discussed there. Deep (low- $b$ ) and broad (high- $b$ ) hierarchies perform differently at different module sizes. For highly granular (small-module-size) projects, a deep hierarchy will often be preferable; for more coarse-

---

grained (large-module-size) projects, a flatter hierarchy will be more efficient. Projects expanding by adding users should take this into account in situating new participants into the existing development hierarchy. By contrast, projects with a stable or predictable hierarchy can take the same result into account by designing the project to have a small or large unit of contribution, as desired.

## 6.2 Consequences for peer-production researchers

Survivorship bias is a phenomenon well-known to those who study organizations—organizations that are in existence are successful in some sense, while those that are unsuccessful disappear from the sample. It is thus difficult to identify characteristics likely to make projects successful, as the appropriate comparison is missing. A theoretical remedy for this bias can be a better understanding of what causes failure in projects, and in the case of peer production this paper can offer some insight.

One result that appears repeatedly in the variations on the model presented in this paper is the existence of an intermediate module size that is “just right” for maximizing productivity. Large deviations from this optimal module size can cause dramatic drops in effective productivity and increases in completion time. In the real world of peer-production projects, this is likely to manifest not as projects that continue at low efficiency, but rather as projects that wither and die due to attrition. Participants become frustrated at the difficulty of making progress, and abandon it for other projects. These low-productivity project designs, then, are one characteristic that failed projects may have in common. Projects observed to be successful are likely to have modular designs closer to the optimum. The discussion above highlighted some notable examples of project failure due to inappropriate modularity. In a forthcoming companion paper, I use data from SourceForge, containing both surviving and abandoned open-source software projects, to further characterize empirically the distinguishing features of successful and unsuccessful endeavors.

## 6.3 Endogeneity of project design

In the analytical and simulation models discussed above, I take as static a number of parameters related to the project’s design. For example, many of the above results concern the project’s module size  $j$ . Similarly, I discuss the hierarchical project structure, characterized by a static  $b$ . One limitation of this approach is the impossibility of accounting for changes in these parameters over

---

time. Changes in  $j$ , for example, correspond roughly to modularizations of the code—changes in the division of the project in to modular units without changing the overall functionality. This process is known to programmers as *refactoring* (Opdyke, 1992), and is not undertaken lightly—while development on individual modules is part of a “steady-state” software-production process, refactoring often requires a complete cessation of other work on the project.

The difficulty of refactoring suggests an analogy with the notion of compatibility standards (David, 1985; David and Greenstein, 1990), which dictate the method of interoperation of products that depend on each other. An example of such a standard would be the shape and mode of use of universal serial bus (USB) cables, which connect computers to keyboards, printers, cameras, and other peripheral devices. Though these components may be produced by diverse manufacturers, conformance to the relevant standards increase the usability of their products. Voluntary commitment by manufacturers to such standards is in contrast to economists’ notion of product differentiation, but is necessary in an environment of modular products. David’s work shows, among other things, how established compatibility standards can be difficult to displace, even when better alternative standards exist. Just so, in a project of modular components, the specification of how these are intended to interact is essential to the functioning of the project as a whole. Though the analogy with compatibility standards may not hold in how a particular modularization is arrived at, departing from an established project design can be similarly difficult.

Though a full economic analysis of open-source production over time would require consideration of project design as endogenous, this is not attempted here. Rather, the analysis contained in this paper should be thought of as static, applying only over certain periods of time. These periods, characterized by “normal” software-development activity, may be punctuated by moments of discrete project redesign, during which the models’ parameters may change dramatically, and after which the predictions of the model can be expected to change correspondingly. While the model may describe project development over individual periods of stable development, it takes no account of the fact that project redesign may very well be undertaken precisely *because* of unsatisfactory outcomes stemming from poor choices of project-design parameters.

Thus, a project with initially very high module size  $j$  may interrupt steady-state development—likely to be very slow, by the results obtained in Section 3.3 above—in order to restructure module interdependence to decrease  $j$ . Inversely, projects that happened on a near-optimal initial choice of  $j$  might never find it necessary to undertake a refactoring.

## 6.4 Methodological choices

This paper introduced a simple theoretical model of modular production and developed in analytically and through simulation. I have argued that the model is a productive metaphor for certain parts of the phenomenon of peer production, and therefore that we can draw conclusions from it about that subject. Like any model, however, it has certain shortcomings and simplifications, which must limit its ability to describe the phenomenon of interest. In this section, I discuss these limitations and their capacity to affect or bias my conclusions.

Microeconomists may be sensitive to the absence of explicit modeling of project participants' incentives. By specifying a structure of incentives and information, this argument claims, a better picture of the interaction of the behavior of many individuals can be obtained. My model, in which these features are absent, by contrast is unable to adequately capture the effect participants have on each other's behavior. This may have effects in several ways. For example, which participants select into the project in the first place is very important for outcomes. An increase in module size, for instance, may cause less-skilled programmers to select out of the project, increasing efficiency by removing the least-efficient tranche of participants.

There is no doubt that these concerns are relevant; participants' incentives are certainly a consideration in real peer-production systems. I have chosen, however, to leave explicit consideration of incentives outside of my model, for two related reasons. First, participants incentives do not appear to be simple or uniform. Ethnographic work such as that of Kely (2008) attests to the diversity of motives for participation in open-source projects. As communication costs decrease, as technological innovations such as wiki software lower barriers to participation, and as large peer-production systems achieve success sufficient to lead to so-called network effects<sup>19</sup>, participation can be expected to become still more diverse. As alluded to above, successful open-source projects not to align individuals' incentives, but rather to find ways to succeed despite the diversity of incentives. This is the aspect of peer production I have chosen to study—how successful projects incorporate the contributions of individuals with diverse incentives.

Still, one might argue that the model studied here could be improved by at least an attempt at explicit treatment of incentives, and one could imagine alternatives to the random contributions that I assume. Within the simulation methodology I have employed, some effort along these lines should

---

<sup>19</sup> This is economists' usual term for goods or services whose value increases as more individuals adopt them, such as Blu-ray high-definition optical disks. Sociologists' usual meaning of "network" is not implied by the use of the term.

---

be achievable, even for large numbers of individuals. However, in studying peer production I hope to offer results relevant both to pure open-source projects (such as Apache and Wikipedia) and to highly modular commercial projects (such as reCAPTCHA and Mechanical Turk). This points to the second reason I have avoided explicit modeling of incentives, *viz.*, that since projects may or may not involve wages or other remuneration, there is no obvious theoretical proxy for participants' goals. While Lerner and Tirole (2002) make a case for a career-concerns expected return to participation, this simply does not seem to be what motivates all participants, nor is it relevant to commercial peer-production projects. Any explicit analytical model of participants' incentives, therefore, runs a risk of being precise at the expense of relevance.

For a methodological parallel, one could look to the work of Farmer et al. (2004) on “zero-intelligence agents” in financial markets. Such agents are assumed, as are agents in the model presented in this paper, to act randomly. The rules of the continuous double auction, a part of the institutional makeup of many financial markets, together with this random behavior, is sufficient to give certain predictions about the behavior of prices. Also in this lineage is the work of Becker (1962), who shows that a downward-sloping aggregate demand curve is implied by the budget constraint alone, even if participants have diverse and perhaps non-utility-maximizing consumption-decision rules. These two studies, like the present one, could be said to describe macro-level features that arise from institutional structure, and that are to some extent independent of specific micro-level behavior. In the presence of the diverse and uncertain incentives of peer production, such an approach seems wise.

## 6.5 Finally

This work is a theoretical investigation of the interaction of project design and organizational hierarchy, undertaken with the intention of applying its results to peer production. Peer production presents a challenge to economics: coordination among large numbers of participants is achieved without the use of prices, wages, or managerial control. Yet, as the process unquestionably constitutes “production”, an understanding of the mechanisms and institutions involved in achieving that coordination is within the province of economics. I have presented here a simple model of a production process that incorporates several features relevant to peer production; naturally, much work remains to be done.

## References

- Gary S. Becker. Irrational behavior and economic theory. *The Journal of Political Economy*, 70(1): 1–13, February 1962. ISSN 00223808. URL <http://links.jstor.org/sici?sici=0022-3808%28196202%2970%3A1%3C1%3A1BAET%3E2.O.CO%3B2-J>.
- Yochai Benkler. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. Yale University Press, New Haven, 2006.
- F.P. Brooks. *The Mythical Man-Month: Essays on Software Engineering*. Addison–Wesley, Reading, 1975.
- Paul A. David. Clio and the economics of QWERTY. *The American Economic Review*, 75(2): 332–337, 1985. URL <http://www.jstor.org/stable/1805621>.
- Paul A. David and Shane Greenstein. The economics of compatibility standards: an introduction to recent research. *Economics of Innovation and New Technology*, 1(1 & 2):3–41, 1990. doi: 10.1080/10438599000000002.
- J. Doayne Farmer, Paolo Patelli, and Ilija I. Zovko. The predictive power of zero intelligence in financial markets, February 2004. URL <http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0309233>.
- Karl Fogel. *Producing open source software: how to run a successful free software project*. 2005.
- Victor M. González and Gloria Mark. Managing currents of work: Multi-tasking among multiple collaborations. In *ECSCW 2005: Proceedings of the Ninth European Conference on Computer-Supported Cooperative Work, 18-22 September 2005, Paris, France.*, pages 143–162, 2005.
- Dan Hunter and John Quiggin. Money ruins everything. *Hastings Communications and Entertainment Law Journal*, 30, 2008.
- Christopher M. Kelty. *Two bits: the cultural significance of free software*. Experimental Futures. Duke University Press, Durham, 2008.
- Bruce Kogut and Anca Metiu. Open-source software development and distributed innovation. *Oxford Review of Economic Policy*, 17(2):248–264, June 2001. ISSN 0266-903X. URL <http://ejournals.ebsco.com/direct.asp?ArticleID=412AA4946E003AAD4194>.
- Richard N. Langlois. Modularity in technology and organization. *Journal of Economic Behavior and Organization*, 49:19–37, 2002.
- Josh Lerner and Jean Tirole. Some simple economics of open source. *Journal of Industrial Economics*, 50(2):197–234, June 2002.
- Josh Lerner, Parag A. Pathak, and Jean Tirole. The dynamics of open-source contributors. *The American Economic Review*, 96(5):114–118, May 2006.
- Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, New York, 1995.
- Siobhán O’Mahony. Guarding the commons: how community managed software projects protect their work. *Research Policy*, 32:1179–1198, 2003.
- William F. Opdyke. *Refactoring object-oriented frameworks*. PhD thesis, University of Illinois at Urbana–Champaign, Urbana, Illinois, 1992.

- 
- Roy Radner. The organization of decentralized information processing. *Econometrica*, 61(5):1109–1146, September 1993. ISSN 0012-9682. URL <http://links.jstor.org/sici?sici=0012-9682%28199309%2961%3A5%3C1109%3AT00DIP%3E2.O.CO%3B2-6>.
- Herbert A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6):467–482, December 1962. ISSN 0003-049X. URL <http://links.jstor.org/sici?sici=0003-049X%2819621212%29106%3A6%3C467%3ATA0C%3E2.O.CO%3B2-1>.
- Adam Smith. *The Wealth of Nations*. Bantam Classics, 1776.
- Clive Thompson. Open-source spying. *The New York Times*, December 2006. URL <http://query.nytimes.com/gst/fullpage.html?res=9406EEDD103EF930A35751C1A9609C8B63&sec=&spon=&partner=permalink&exprod=permalink>.
- Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008. doi: 10.1126/science.1160379. URL <http://www.sciencemag.org/cgi/content/abstract/321/5895/1465>.
- Eric von Hippel. *Democratizing Innovation*. The MIT Press, Cambridge, 2005.
- Eric von Hippel. *The sources of innovation*. Oxford University Press, New York, 1988.
- Jimmy Wales. Let’s make a wiki. Nupedia-l mailing list, January 2001. URL <http://web.archive.org/web/20030414014355/http://www.nupedia.com/pipermail/nupedia-l/2001-January/000676.html>.
- Dennis M. Wilkinson. Strong regularities in online peer production. In *Proceedings of the 2008 ACM Conference on E-Commerce*, Chicago, IL, July 2008.