Optimal Task Assignments*

Steve Yeh†

October 22, 2024

Abstract

A principal assigns tasks to her team of agents working on a project with multiple tasks. Effort spent working on a task is a strategic substitute for effort spent finishing related tasks and is subject to a convex cost. Agents are prone to shirking and will free-ride off other agents' efforts. Assigning agents to more tasks mitigates free-riding, but could overload them and prevent them from finishing all of their assigned tasks. I focus on a simple class of projects called key-peripheral projects composed of peripheral tasks (related to one other task) and key tasks (related to at least two peripheral tasks). The modular assignment, which assigns each agent to a module consisting of a key task and its peripheral tasks, is optimal if every key task is sufficiently specialized. A key task exhibits high specialization if linked to more peripheral tasks than key tasks. Characterizing the optimal task assignment beyond key-peripheral projects requires additional assumptions on the cost of effort and project structure.

*I am indebted to Evan Sadler for his encouragement, guidance, and careful readings. I thank Melanie Chen, Zhiming Feng, Navin Kartik, Jacopo Perego, and audiences at the 35th Stony Brook International Conference on Game Theory and Columbia Microeconomic Theory Colloquium for their helpful comments.

†Columbia University— sy3012@columbia.edu.

1 Introduction

Task assignment is a fundamental managerial responsibility. It determines team member workloads and influences how team members interact and coordinate with each other on a project. Understanding how assigning tasks shapes team productivity is crucial because, in most organizations, managers cannot access a well-studied tool—contract design. Instead, a separate compensation and benefits department designates employee salaries and bonus rates. In this paper, I study how task assignments affect project outcomes.

How a manager assigns tasks should depend on the underlying project task structure. Consider a software development team building an application where developers can reference code written by others working on similar tasks. A developer working on a feature related to many complementary features may shirk and *free-ride* off others' efforts. Unfortunately, multiple developers coding those complementary features need to increase their efforts to compensate for one developer's free-riding behavior. To counteract free-riding, the manager can assign each developer to more tasks. Doing so encourages them to internalize their externalities. However, assigning a developer too many tasks could *overload* him: he will tire quickly and fail to complete all of his assigned tasks. In light of this trade-off, how should the principal optimally assign tasks? How does the project architecture affect the structure of the optimal task assignment?

I answer these two questions by modeling a project as a network of tasks. A link between two tasks indicates effort exerted at one task is a strategic substitute for effort at the other task. The main novelty of the model is that a principal assigns agents to tasks. Each agent exerts efforts at assigned tasks subject to a convex cost and finishes a task if his effort together with the sum of efforts at linked tasks reaches an exogenous completion threshold. An agent is evaluated by the number of tasks he completes, while the principal seeks to complete as many tasks in the project as possible while keeping total team effort costs low.

The model entails a simple binary evaluation of task completion, keeping the analysis tractable and practical. For instance, progress made towards fixing a piece of software that crashes is futile if the principal needs it to be fully debugged for deployment. Cleaning half a dataset is impractical; the principal needs the whole dataset scrubbed to uphold the integrity of related analyses. The principal's objective captures the fact that managerial evaluations often assess how well the manager builds, supports, and moti-

vates her team (Hyväri (2006); Chen and Lee (2007)). Managing team effort costs is critical to a favorable performance review for the principal because they measure fatigue, a key contributor to employee burnout and low morale (Maslach and Leiter (2016)).[1] I assume that agents are identical and tasks have identical completion thresholds. Suppressing skill-matching incentives for the principal allows us to demonstrate how the optimal task assignment depends on the project architecture. I also assume there are more agents than tasks to eliminate understaffing concerns.

In the main analysis, I focus on the class of *key-peripheral projects* composed of *peripheral tasks* that are linked to only one other task and *key tasks* that are linked to at least two peripheral tasks. Key-peripheral projects include a variety of commonly studied networks such as stars, connecting stars, and hierarchical trees with at least two leaves at every branch. They can model simple data analysis projects, where preparing a dataset is a key task, and analyses that use different subsets of the data are peripheral tasks.[2] Moreover, key-peripheral structures model software development projects particularly well because they embody a basic design mechanism— inheritance. Inheritance champions code reuse in constructing parent classes and methods so that subclasses and auxiliary methods can reference them when building specific functionalities.[3] Creating parent classes and building general libraries of functions are key tasks while coding accessory methods and features correspond to peripheral tasks. Excluding classes and methods that are neither key nor peripheral avoids adding unnecessary design complexity and depth of inheritance that make it difficult to predict how the software behaves (see Chidamber and Kemerer (1994) and Subramanyam and Krishnan (2003)).

Characterizing the optimal task assignment depends on how the trade-off between mitigating free-riding and task overload manifests in key-peripheral projects. The primary object of analysis borrows from a familiar concept in organizational economics: modularization.[4] The *modular assignment*, which assigns each agent to a *module* con-

[1] A 2022 Forbes article reported 67% of survey respondents experiencing some level of burnout resulting from feeling overworked.

[2] Free-riding takes the following form: team members clean sections of the dataset relevant to their respective analyses. The analyst in charge of preparing the whole dataset free-rides and just compiles the various cleaned subsets of the data together.

[3] See IBM's primer on inheritance. Figure 4 of the guide displays the code architecture for an elementary bank account system, which is key-peripheral. While the direct network illustrates inheritance relationships, effort substitutability between related coding tasks is bidirectional.

[4] The idea of modularization was first popularized by Simon (1965) and has become a keystone concept in software development and production design. Parnas (1972) and Garud, Langlois and Kumaraswamy (2003) describe the history and prevalence of modularization in software development and production design.

sisting of a key task and all of its offshoot peripheral tasks, serves as a simple yet powerful heuristic. It performs remarkably well with regard to implementability and optimality.

The first main result is that the modular assignment implements an equilibrium where agents exert full effort to reach the completion threshold at key tasks and zero effort at peripheral tasks (Theorem 1). In particular, this equilibrium is unique. The modular assignment resolves the free-rider problem: agents do not shirk at key tasks, even though they may be linked to other key tasks where other agents play full effort. The modular design also prevents task overload by assigning agents to tasks that are related to each other, enabling them to effectively internalize their externalities and avoid overexertion.

That the equilibrium implemented by the modular assignment takes a specific graph theoretic structure allows us to extend implementability to arbitrary projects. The equilibrium is a *minimum dominating effort profile*: tasks with full effort form a *minimum dominating set* of the project and every other task in the project has zero effort. A *dominating set* is a set of tasks where every task in the project is either in that set or linked to a task in that set; a minimum dominating set is a dominating set of minimum size. Minimum dominating effort profiles admit a simple structure and complete every task in the project. I show that any minimum dominating effort profile can be implemented as an equilibrium by a *generalized* modular assignment (Theorem 2). Any other equilibria implemented by a generalized modular assignment obtain a principal's payoff no worse than a minimum dominating effort profile (Theorem 3). These results together establish a lower bound to what a principal can accomplish when facing an arbitrary project.

The second main result is that the modular assignment is optimal for key-peripheral projects with sufficiently specialized key tasks (Theorem 4). The *specialization* of a key task is the ratio of its number of peripheral tasks to the number of key tasks it is linked to plus one. Specialization neatly captures the trade-off between mitigating free-riding and task overload. To see this, first note that while the modular assignment completes every task, each agent pays a relatively high effort cost. A better task assignment must reduce team effort costs by *underloading* agents: split up a module by assigning a different agent to exactly one task within the module and implement an equilibrium where each agent completes his assigned task by exerting less than full effort. As expected, doing so reintroduces free-riding and the agent assigned to the key task will inevitably shirk. It turns out the only way to support this scheme in equilibrium is to assign him a neighboring key task and leave every peripheral task of that key task incomplete.

Connecting this back to the project structure, if every key task is highly specialized i.e. linked to many peripheral tasks, then leaving every peripheral task of a key task is suboptimal. Thus key tasks are sufficiently specialized only if the modular assignment is optimal.

High key task specialization is analogous to the concept of *loose coupling* in systems design, which entails a low degree of interdependence between modules within a system. Projects with specialized key tasks/loosely coupled modules are common and possess several appealing properties: easy testability, simple scalability, reduced maintenance costs, and increased immunity to catastrophic cascading failures (Stevens, Myers and Constantine (1999)).

For projects with low key task specialization, there is no definite characterization of the optimal task assignment. Underloading, as described before, might be beneficial. The principal can use the key task with low specialization to reduce effort costs at multiple neighboring modules while leaving relatively few peripheral tasks of said key task unfinished. Whether this achieves a higher principal's payoff than the modular assignment hinges on the shape of the effort cost. Formally, I show if a key-peripheral project contains a key task with specialization below one, then there exists an effort cost function that makes the modular assignment optimal and another effort cost function that makes it suboptimal (Theorem 5).

Moving beyond key-peripheral projects, I study generalized modular task assignments on a broader class of projects that consist of key, peripheral, and intermediary tasks. Intermediary tasks are helper tasks that aid progress at multiple key tasks but are not linked to any peripheral tasks. An analogous null result regarding optimality applies: for any project composed of key, peripheral, and intermediary tasks, there exists an effort cost function that makes the generalized modular assignment optimal and another effort cost function that makes it suboptimal (Theorem 6). All together, this establishes a boundary about the main result. Key-peripheral projects with highly specialized key tasks constitute a large class of interpretable projects that admit a simple characterization of the optimal task assignment *without* additional assumptions on the project structure and effort cost.

To be transparent, I make two substantive assumptions: perfect substitution of effort between linked tasks and no staffing shortages. Relaxing the first assumption, I show that the modular assignment is optimal (given sufficiently specialized key tasks) so long as the degree of effort substitutability between tasks is sufficiently high (Theorem 7).

Robustness of the main result is a critical property because related tasks, in practice, may exhibit moderate to high levels of effort substitutability, but not perfect substitutability. With regards to understaffing, the main optimality result still holds if the number of agents weakly exceeds the number of key tasks as the modular assignment remains feasible. The analysis remains inconclusive when the team size drops below that number. Whether it is profitable to assign each agent more tasks and overwork them or assign each agent fewer tasks and leave tasks incomplete again depends on the specific project structure and effort cost.

1.1 Related Literature

This paper contributes to the literature on network games, principal-agent problems, and incentive design for teams.[5] In the networks literature, agents are typically treated as nodes in the network. That notion is decoupled in my model: tasks make up the network, and a principal assigns agents to tasks. The set-up of my model is most similar to Bramoullé and Kranton (2007), who study the provision of a public good that cannot be excluded along links in a network. Equilibria in their model feature free-riding and take the shape of maximal independent sets of the network: agents who exert full effort do not neighbor each other. Introducing task assignments expands the set of equilibria to include minimum dominating effort profiles. Crucially, minimum dominating sets do not have to be independent sets. Many recent papers focus on public good provision in networks and discuss the free-rider problem (see Allouch (2015); Kinateder and Merlino (2017, 2023); Elliott and Golub (2019)). In more general analyses, Bramoullé, Kranton and D'Amours (2014) determine the structure of all Nash equilibria in network games that exhibit linear best responses. To tackle the prevalent issue of equilibrium multiplicity, Galeotti et al. (2010) incorporate incomplete information into network games. In a setting with complete information, task assignments can resolve the issue of equilibrium multiplicity. In particular, I show that the modular task assignment implements a unique equilibrium.

Studying optimal contracts with moral hazard is common in research on principal-agent problems, starting with the classic papers of Mirrlees (1976) and Holmström

[5] Assignment problems are ubiquitous in operations research, but the treatment does not allow agents to behave strategically nor incorporate incentives and equilibrium analysis. Ahuja, Magnanti and Orlin (1993) provides a comprehensive overview of classical assignment problems. Approaches to solving assignment problems include rewriting them as network flow problems or graph coloring problems.

(1979). Many variations and extensions have been considered, such as adding a retention rule (Banks and Sundaram (1998)), incorporating dynamics (Holmström (1999)), and endogenizing the information acquisition process (Georgiadis and Szentes (2020)). Holmström (2017) provides an overview of the literature. To pin down how the project structure affects the principal's problem, I do not incorporate uncertainty in my environment. Instead of writing contracts or monitoring agents, the principal can only assign tasks. Task assignments resemble delegation, where the principal delegates a set of decisions to the agent from which he makes a choice (e.g. Alonso and Matouschek (2008)). In my model, the decisions an agent faces are linked to each other by way of the underlying project architecture. In Matouschek, Powell and Reich (2024), a principal designs communication networks for agents trying to match their actions to exogenous local states and coordinate with other agents residing within the same module of production. The authors focus on how a firm's organizational structure, specifically a modular production network, affects the structure of the optimal communication network. In contrast, the principal in my model chooses a particular task assignment and influences the production design rather than taking it as fixed.

The literature on contract design for teams similarly focuses on environments with moral hazard and conflicts of interest (e.g. Groves (1973); Holmström (1982); Che and Yoo (2001); Georgiadis (2015)). Recent work focuses on how a manager with limited commitment chooses project objectives for a team of agents (Georgiadis, Lippman and Tang (2014)), determines the optimal team size when the project is risky (Fu, Subramanian and Venkateswaran (2016)), finds the optimal team composition (Glover and Kim (2021)) and characterizes the optimal incentive structure for agents with production spillovers (Dasaratha, Golub and Shah (2024)). I contribute to this literature by studying how a manager should optimally assign tasks to a team of agents where task collaboration is determined by the project structure— the optimal assignment is the modular assignment given sufficiently high key task specialization.

2 Model

$N = \{1, ..., n\}$ is a set of agents and $K = \{1, ..., k\}$ is a set of tasks where $n \geq k$.[6] A project is a network of tasks G where links between tasks indicate perfect effort substitutability. The principal chooses a task assignment such that no more than one agent is assigned to a task. Agents choose how much effort to exert at their assigned tasks. Formally, the principal chooses $f : N \to 2^K$ from the set $F = \{f \mid \forall a, d \in N, f(a) \cap f(d) = \emptyset\}$ which induces an effort game played by agents $\langle M, ((\mathbb{R}^+)^{|f(a)|})_{a \in M}, (\pi_a)_{a \in M} \rangle$:

- $M \subseteq N$ is the set of assigned agents i.e. agents a with $f(a) \neq \emptyset$.
- $(\mathbb{R}^+)^{|f(a)|}$ is the effort space for agent a. He chooses effort levels at each task in $f(a)$; denote e_i as the effort chosen at task i.
- $\pi_a = \sum_{i \in f(a)} \mathbb{1}\{e_i + \sum_{\ell \in G_i} e_\ell \geq b\} - c(\sum_{i \in f(a)} e_i)$ is the payoff to agent a where:
  - b is an exogenous task completion threshold.
  - G_i is the set of tasks that neighbor task i in network G.
  - $c : \mathbb{R}^+ \to \mathbb{R}^+$ is a C^1, strictly increasing and strictly convex function with $c(0) = 0$, $c'(0) = 0$ and $c(b) < 1$.

Agents are evaluated by the number of assigned tasks they complete. Task i is completed if and only if it is assigned and there are at least b units of effort exerted. Efforts expended at linked tasks in G_i contribute to completing task i. An agent's net payoff is the number of assigned tasks completed less his cost of effort, where the latter is a function of his aggregate effort. The cost of full effort $c(b)$ is strictly less than one. In other words, for an agent assigned to exactly one task, his best response is to work just enough to reach the completion threshold. Agents who remain unassigned receive a normalized payoff of zero. Tasks can also remain unassigned, have zero exerted effort, and cannot be completed.

Given a task assignment, agents play a pure strategy Nash equilibrium in the induced effort game. Every agent chooses effort levels to maximize his payoff in response to the effort provisions of all other agents. The principal chooses a task assignment $f \in F$ such that an induced pure strategy Nash equilibrium maximizes the total payoff across

[6] For now, there are at least as many agents as tasks. Whether the principal can implement a certain outcome depends only on the agents' strategic behavior and not on understaffing concerns. This assumption will be relaxed in Section 7.2.

the entire project, which is the total number of tasks completed less the total effort costs. There may be multiple Nash equilibria for some assignments; I assume that the equilibrium that maximizes the principal's payoff is selected. Formally, the principal's problem is:

$$\max_{f \in F} \sum_{i \in \bigcup_{a \in M} f(a)} \mathbb{1}\{\hat{e}_i + \sum_{\ell \in G_i} \hat{e}_\ell \geq b\} - \sum_{a \in M} c\left(\sum_{i \in f(a)} \hat{e}_i\right)$$

where $\{\hat{e}_i\}_{i=1}^k$ is a pure strategy Nash equilibrium induced by task assignment f.

Remark 1. Since a task is completed only if it is assigned and unassigned agents receive zero payoff, the principal's objective can be rewritten as:

$$\max_{f \in F} \sum_{a \in N} \left(\sum_{i \in f(a)} \mathbb{1}\{\hat{e}_i + \sum_{\ell \in G_i} \hat{e}_\ell \geq b\} - c\left(\sum_{i \in f(a)} \hat{e}_i\right)\right)$$

The principal is utilitarian as she maximizes the total welfare of agents. Abandoning a utilitarian perspective, a principal may instead care only about the number of completed tasks. The problem becomes trivial as the principal is no longer concerned with free-riding behavior, which affects the team effort costs. Assigning each agent to exactly one task (until there are no more tasks left) is a solution to that problem since each agent's best response is to complete their task by assumption of $c(b) < 1$.

3 Key-Peripheral Projects

The main analysis focuses on the class of key-peripheral projects.

Definition 1. A **peripheral task** is linked to only one other task. A **key task** has at least two neighboring tasks that are peripheral and the set of all peripheral tasks of a key task is its **peripheral set**. A **key-peripheral** project contains only key and peripheral tasks.

An agent spending effort on a key task expedites progress at many other auxiliary tasks. Peripheral tasks are specialized offshoot tasks— they synergize well with only one other task in the project. Key-peripheral projects, which contain only key and peripheral tasks, model simple task structures. It is a flexible and interpretable class of networks that includes stars, connecting stars, and trees where every branch (including the root) has at least two leaves. Note that the definition of key-peripheral projects does

not restrict how key tasks are linked to each other. In Figure 1, the square key tasks comprise a cycle.

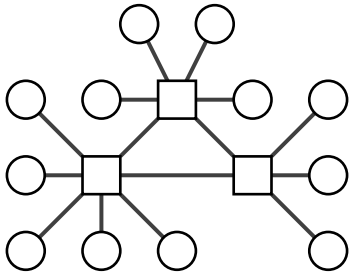


Figure 1: A Key-Peripheral Project

Key-peripheral projects exclude intermediary tasks that facilitate progress at key tasks but do not have offshoot tasks themselves. Figure 2 depicts a network that contains an intermediary triangle task which is neither key nor peripheral. I show in Section 6.3 that characterizing optimal task assignments for projects that are not key-peripheral depends on the particular project structure and the shape of the effort cost.

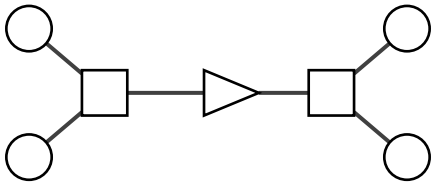


Figure 2: A Non-Key-Peripheral Project

4 Free-Riding versus Task Overload

I illustrate the central tension in the model between free-riding and task overload with a simple example. The payoff to agent a assigned to tasks in $f(a)$ is $\sum_{i \in f(a)} \mathbb{1}\{e_i + \sum_{\ell \in G_i} e_\ell \geq \frac{6}{7}\} - (\sum_{i \in f(a)} e_i)^2$. The completion threshold is $\frac{6}{7}$ and the cost of effort is quadratic. Consider the connecting star, which is key-peripheral. The key tasks are square nodes.

(a) Problem: Free-Riding (b) Problem: Task Overload

Figure 3: Connecting Star

Each color represents a different assigned agent. I say an assignment **implements** an effort profile if it is a Nash equilibrium in the assignment-induced effort game. The left assignment is the trivial assignment which assigns each agent to one task. The principal would hope agents assigned to key tasks exert high efforts to alleviate effort costs for multiple agents working on linked peripheral tasks. The key tasks are linked to each other, so unfortunately, it cannot be that *both* key task agents exert high effort in equilibrium. Any equilibrium implemented by the trivial assignment exhibits **free-riding** where at least one agent assigned to a key task shirks and agents assigned to corresponding peripheral tasks play high efforts. Free-riding is harmful to the principal as it drives up team effort costs. To alleviate free-riding, the principal can assign every task to one agent, as she does in the right assignment. Unfortunately, the blue agent suffers from **task overload**: he cannot complete every assigned task due to convex effort costs. Assigning the whole project to one agent may leave tasks incomplete.

Figure 3 depicts the two "extreme" assignments, and the optimal task assignment intuitively should balance mitigating free-riding and task overload. The modular assignment, as defined next, strikes this balance.

Definition 2. A **module** is a set of tasks containing a key task and its peripheral set. On key-peripheral projects, a **modular assignment** assigns each agent to one module within the project (until all modules are assigned).

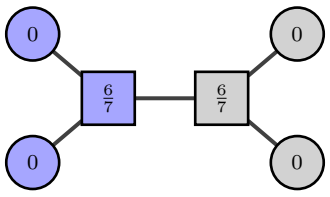


Figure 4: Equilibrium via a Modular Assignment

Modularization is a simple yet powerful heuristic. The modular design obliges each agent to internalize enough of his externalities so that he always exerts high effort at his key task, thereby resolving the free-riding problem. While agents are assigned multiple tasks, they are not assigned dissimilar tasks. Agents effectively exploit effort substitutability among their assigned tasks, avoiding task overload and high convex costs. In the main analysis, I focus on the implementability and optimality of the modular assignment.

5 Implementability

One of the appealing properties of the modular assignment is its unique implementability.

Theorem 1. *On key-peripheral projects, the modular assignment implements a Nash equilibrium with full effort at key tasks and zero effort everywhere else. In particular, this equilibrium is unique.*

Proof. The modular assignment assigns each agent to a module. Each assigned agent plays effort b at his assigned key task i and zero effort at every task in the peripheral set S_i in equilibrium. Assume by contradiction that he chooses some other effort level $e_i \in [0, b)$. Then playing effort $e_h = b - e_i$ at one of his peripheral tasks h ensures the completion of two tasks (i and h) at a cost of $c(e_h + e_i) = c(b)$. But $|S_i| \geq 2$, so there is at least one other assigned task that is left incomplete. Setting $e_i = b$ completes more tasks with a weakly lesser cost. Since each agent's optimal effort provision is independent of every other agent's effort provision, this is the unique equilibrium. □

The modular assignment rectifies the equilibrium multiplicity problem that is frequently encountered in network games.[7] The unique equilibrium takes a particular graph theoretic structure— a minimum dominating effort profile. This language allows us to extend implementability to arbitrary projects.

Definition 3. For a graph G and a subset D of the vertex set $V(G)$, let $G[D]$ be the set of vertices of G which are in D or adjacent to a vertex in D. If $G[D] = V(G)$, then

[7]The unique equilibrium can be implemented by assigning each agent to a key task and two peripheral tasks instead of the entire peripheral set. Nevertheless, assigning agents to the entire peripheral set matters for the extension of imperfect substitutability. As I will show in Section 7.1, assigning agents to more peripheral tasks within their module allows lower levels of effort substitutability to support optimality.

D is a **dominating set** of G. A dominating set of the smallest size is a **minimum dominating set**; the size of a minimum dominating set is called the **domination number** $\gamma(G)$.[8] A **minimum dominating effort profile** prescribes effort b at tasks in a minimum dominating set and zero effort everywhere else.

Minimum dominating effort profiles are attractive for two reasons. First, they exploit effort substitutability and complete every task in the project with positive effort exerted at the least number of tasks. Second, minimum dominating effort profiles are implementable on *any* project structure via a *generalized* modular assignment.

Theorem 2. *Every minimum dominating effort profile of a project G with k tasks is implementable with a principal's payoff of $k - \gamma(G)c(b)$.*

Proof. See Appendix. □

Agents assigned to tasks residing in the underlying minimum dominating set that are linked to each other will not all play full effort; some will free-ride. To prevent this, the principal can use a *generalized* modular assignment.

Algorithm 1 Generalized Modular Assignment.

Require: D_{min} a minimum dominating set of graph G.

1: **for** each task $i \in D_{min}$ **do**
2: Assign task i and every unassigned $j \in G_i \setminus D_{min}$ to an unassigned agent.
3: **end for**

A generalized modular assignment assigns agents to a task in a minimum dominating set *and* extra linked tasks whenever possible. The additional assigned tasks ensure agents internalize their local externalities by exerting full effort at their task in a minimum dominating set. Since each agent is exerting effort b and there are $\gamma(G)$ assigned agents (one for each task in a minimum dominating set), the net payoff to the principal is $k-\gamma(G)c(b)$. Algorithm 1 is agnostic of the order in which we assign tasks in a minimum dominating set, so there may be multiple generalized modular assignments that implement a particular minimum dominating effort profile. Figure 5 depicts a generalized modular assignment on a non-key-peripheral project where the square tasks constitute a minimum dominating set of the project.

[8] Dominating sets are well-studied by graph theorists. For further reference, see West (2001). In general, finding a minimum dominating set (and therefore the domination number) is an NP-hard problem where a brute force algorithm has a time complexity of $O(2^n)$. On key-peripheral networks, the unique minimum dominating set is the set of key tasks.

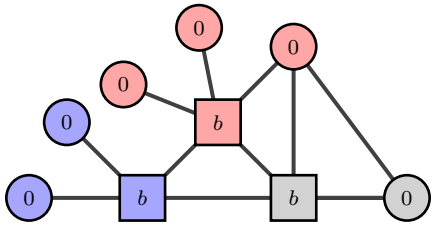


Figure 5: Generalized Modular Assignment

Now, a generalized modular assignment (corresponding to some minimum dominating effort profile) may not implement a unique equilibrium. Fortunately, every other equilibrium implemented by a generalized modular assignment achieves a principal's payoff that is no worse than that of a minimum dominating effort profile.

Theorem 3. *For a project G, every equilibrium implemented by a generalized modular assignment (corresponding to a minimum dominating effort profile) achieves a principal's payoff weakly greater than $k - \gamma(G)c(b)$.*

Proof. Assume by contradiction that an equilibrium implemented by a generalized modular assignment achieves a principal's payoff strictly less than $k-\gamma(G)c(b)$. Either some task is left incomplete or the team incurs a higher total effort cost. If the former were true, then there is some agent who has an assigned task that is left incomplete because every task is assigned by a generalized modular assignment. That agent has a profitable deviation by playing b at his task residing in the minimum dominating set since he increases his payoff by at least $1 - c(b) > 0$. Thus the team must incur a higher total effort cost, implying some agent exerts strictly more effort than b among their assigned tasks. But he has a profitable deviation by just playing effort b at his task in the minimum dominating set, which saves on effort and still completes every one of his assigned tasks. □

This result establishes a lower bound to what the principal can achieve given any project structure independent of an equilibrium selection rule. She can always complete every task in the project and guarantee herself a payoff of at least $k - \gamma(G)c(b)$ for a project G by implementing a minimum dominating effort profile via a generalized modular assignment.

6 Optimality

6.1 Preliminary Analysis: Efficiency

As a preliminary analysis, I derive the efficient assignment and effort profile on key-peripheral networks ignoring agents' strategic behavior. The efficient assignment is the trivial assignment that assigns each agent to one task as it achieves the lowest cost by convexity.[9] The efficient effort profile solves a specific cost-minimization problem:

$$\min_{\{e_i\}_{i=1}^k} \sum_{i=1}^{k} c(e_i) \quad \text{s.t. } e_i + \sum_{\ell \in G_i} e_\ell \geq b \text{ and } e_i \geq 0 \text{ for all } i$$

The effort threshold must be satisfied at each task. If not, then the principal is strictly better off by having full effort level b exerted at the unfinished task, which increases her total payoff by at least $1 - c(b) > 0$. The following proposition fully characterizes the efficient effort profile on key-peripheral networks.

Proposition 1. *The efficient effort at key task i solves $c'(e_i) = |S_i| \cdot c'(b - e_i)$ where S_i is its peripheral set. The efficient effort at a peripheral task $h \in S_i$ is given by $e_h = b - e_i$.*

Proof. See Appendix. □

The efficient effort level at a key task is high, but not full effort. It increases with the size of its peripheral set because the ratio of the marginal costs $\frac{c'(e_i)}{c'(b-e_i)}$ is strictly increasing in e_i due to strict convexity. Efficient peripheral task efforts are low but nonzero as spreading the effort levels across tasks helps keep total team effort costs down. This preliminary analysis provides a glimpse at the advantages of the modular assignment. Notice that the efficient effort level at a key task is independent of the efforts played at other key tasks. The modular assignment achieves exactly that as we saw in Theorem 1.

6.2 Optimal Task Assignment

Finding an optimal task assignment on key-peripheral projects is challenging because computing an equilibrium that maximizes the principal's payoff for a given task assignment can depend on how key tasks are linked to each other, which key tasks have more

[9] A strictly convex cost with $c(0) = 0$ satisfies superadditivity i.e. $c(\sum_{i \in f(a)} e_i) > \sum_{i \in f(a)} c(e_i)$.

peripheral tasks, etc. That there are $(n+1)^k$ possible task assignments, where n is the number of agents and k is the number of tasks, makes the comparing assignments particularly unwieldy. Fortunately, we can use the modular assignment as a benchmark to resolve these difficulties because it completes every task in the project and its principal's payoff is pinned down by two elementary attributes, namely the total number of tasks k and the number of key tasks $\gamma(G)$. The following result dramatically reduces the space of assignments comparable to the modular assignment.

Proposition 2. *If an assignment achieves a strictly higher payoff than the modular assignment, then it must implement effort $e_i \in (0, b)$ at some key task i and leave the entire peripheral set of a neighboring key task $j \in G_i$ unfinished.*

Proof. To achieve a strictly higher payoff than modularization, an assignment must have a lower total cost. To do so, it must implement an effort profile with effort $e_i \in (0, b)$ at some key task i ($e_i = 0$ is strictly worse and $e_i = b$ replicates the modular assignment payoff). The assignment must complete every peripheral task of a key task i with $e_i \in (0, b)$. If not, then the modular assignment achieves a strictly higher payoff since $c(b) < 1$ by assumption.

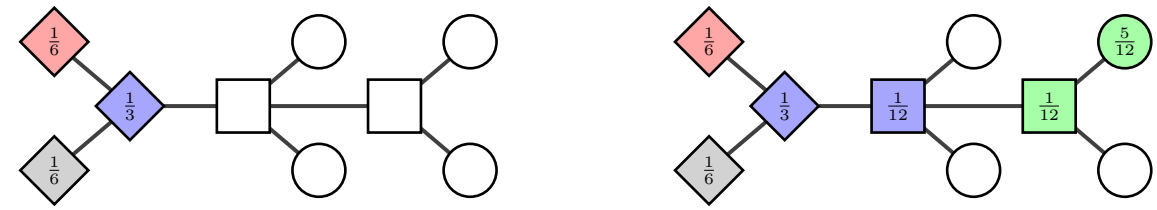
Let S_i denote key task i's peripheral set. To complete every task in S_i, it is without loss to focus on an assignment that assigns each peripheral task in S_i to a different agent who exerts effort $b - e_i$; assigning an agent to more than one peripheral task does not help the principal's payoff because of convex cost superadditivity. The effort threshold at key task i is at least $e_i + |S_i|(b - e_i) > b$, causing the key task agent to shirk. This can be supported as an equilibrium only if the key task agent is also assigned to some neighboring key task $j \in G_i$ that has $e_j + \sum_{\ell \in G_j} e_\ell = b$ in equilibrium. If the key task agent is not assigned to a linked key task, then he will reduce his efforts from e_i at key task i. If he is assigned a neighboring key task j but the completion threshold at j does not bind, then the agent will again have a profitable deviation by decreasing his effort at task i. This implies every peripheral task $\ell \in S_j$ has to be left unfinished in equilibrium since $e_\ell + e_j < e_\ell + e_j + e_i \leq e_j + \sum_{k \in G_j} e_k = b$. □

This result pares down the space of potentially better assignments to ones implementing effort profiles that are *locally* different from the minimum dominating effort profile.[10] A direct implication is that the modular assignment is optimal on projects with disconnected key tasks. Moreover, efficiency is never attainable as the efficient

[10] This trade-off does not always exist for arbitrary networks. See Example 3.

effort profile prescribes effort less than b at key tasks *and* completes every task in the project. In the following example, I illustrate how free-riding and task overload play a role in Proposition 2.

Example 1. Take the key-peripheral project shown below with completion threshold $b = \frac{1}{2}$ and a quadratic cost function $c(e) = e^2$. Consider an alternate assignment that implements a key task effort of $\frac{1}{3}$ on the diamond module:



(a) Reducing Costs on the Diamond Module

(b) An Equilibrium

Figure 6: Alternate Assignment

In Panel (a), every task in the diamond module is completed at a total cost of $2 \cdot (\frac{1}{6})^2 + (\frac{1}{3})^2$, which is lower than the cost of the modular assignment of $(\frac{1}{2})^2$. The three assigned agents are **underloaded**; they exert less than full effort while completing their tasks. However, observe that the sum of efforts at the diamond key task exceeds the completion threshold; the blue agent will always shirk. Reducing effort costs by underloading agents necessarily reintroduces free-riding. The only way to support an effort level of $\frac{1}{3}$ at the diamond key task in equilibrium is to assign him a neighboring key task, which I call a **supporting** key task. Panel (b) depicts one possible equilibrium. The sum of efforts that contribute to completing the supporting key task must be equal to $\frac{1}{2}$, otherwise the blue agent has a profitable deviation by shirking at the diamond key task. This restriction ensures that the peripheral tasks of the supporting key task remain unfinished in equilibrium.

To capture the trade-off presented in Proposition 2 between free-riding and task over/underload, I introduce the specialization of a key task.

Definition 4. The **specialization of key task** i is $\sigma_i = \frac{|S_i|}{|G_i \setminus S_i|+1}$ where S_i is key task i's peripheral set.

The specialization of a key task is the ratio of its number of peripheral tasks to its number of linked key tasks plus one.[11] The modular assignment is advantageous if every

[11]The denominator can be alternatively written as the size of a key task's *closed* neighborhood of key tasks.

key task is highly specialized because reducing costs by underloading agents necessarily leaves large sets of peripheral tasks incomplete. In contrast, if a key task exhibits low specialization, then designating said key task to support underloading agents and reducing effort costs at multiple neighboring modules (while sacrificing its own set of peripheral tasks) could be favorable. This leads to the main result concerning optimality.

Theorem 4. *Let $\mathcal{K}(G)$ be the set of all key tasks in key-peripheral network G. If $\min_{i\in\mathcal{K}(G)} \sigma_i \geq c(b)$, then the modular assignment is optimal. The optimal number of assigned agents is the domination number $\gamma(G)$.*

Proof. Fix an alternate assignment that implements intermediate effort $e_i \in (0, b)$ at some key task i and completes all of its peripheral tasks. Let L be the set of all key tasks with implemented efforts in $(0, b)$ and completed peripheral sets. Let U be the set of key tasks that support intermediate effort at key tasks in L in equilibrium. Since L is nonempty, then U is nonempty and the peripheral sets of key tasks in U are left unfinished in equilibrium by Proposition 2.

It is without loss to compare payoffs for modules with key tasks residing in L and U. The modular assignment does weakly better than the alternate assignment at other modules not in L and U where key tasks have implemented effort of b or 0, or have incomplete peripheral sets. The modular assignment achieves a weakly higher payoff if:

$$\underbrace{\sum_{i\in L\cup U}(|S_i|+1)-(|L|+|U|)\cdot c(b)}_{\text{modular assignment payoff}} \geq \underbrace{\sum_{i\in L}(|S_i|+1)+|U|}_{\text{upper bound on alt. assignment payoff}} \iff c(b) \leq \frac{\sum_{i\in U}|S_i|}{|L|+|U|}$$

Since every key task in L neighbors a key task in U, we can write:

$$L \subseteq \bigcup_{i\in U}(G_i \setminus S_i) \implies |L| \leq \sum_{i\in U}(|G_i \setminus S_i|)$$

The following inequalities hold:

$$\frac{\sum_{i\in U}|S_i|}{|L|+|U|} \geq \frac{\sum_{i\in U}|S_i|}{\sum_{i\in U}(|G_i \setminus S_i|+1)} \geq \min_{i\in U}\frac{|S_i|}{|G_i \setminus S_i|+1} \geq \min_{i\in\mathcal{K}(G)}\frac{|S_i|}{|G_i \setminus S_i|+1} = \min_{i\in\mathcal{K}(G)}\sigma_i$$

If $c(b) \leq \min_{i\in\mathcal{K}(G)}\sigma_i$, then $c(b) \leq \frac{\sum_{i\in U}|S_i|}{|L|+|U|}$ for any L and U of any alternate assignment that satisfies the necessary condition outlined in Proposition 2 to beat the modular assignment. In other words, no assignment achieves a strictly higher payoff than the

modular assignment, proving optimality. The modular assignment assigns as many agents as the number of key tasks, which is the graph domination number $\gamma(G)$. □

The proof formalizes the intuition and trade-off described before about mitigating free-riding and task over/underload. The modular assignment assigns $\gamma(G)$ agents and if the total number of agents n exceeds $\gamma(G)$, then the principal should keep $n - \gamma(G)$ agents unassigned.[12] The next result relates the modular assignment payoff to the efficient payoff.

Proposition 3. *The principal's payoff to the modular assignment converges to efficiency as $\sigma_i \to \infty$ for each key task i in a key-peripheral project (holding the number of key tasks fixed).*

Proof. Recall that the efficient effort at key task i solves $c'(e_i^*) = |S_i| \cdot c'(b - e_i^*)$. The following inequality must hold, where S_i is the peripheral set of key task i:

$$\underbrace{\sum_{i \in \mathcal{K}(G)} |S_i| + 1 - \sum_{i \in \mathcal{K}(G)} c(e_i^*) - \sum_{i \in \mathcal{K}(G)} |S_i| c(b - e_i^*)}_{\text{efficiency}} \geq \underbrace{\sum_{i \in \mathcal{K}(G)} |S_i| + 1 - \gamma(G)c(b)}_{\text{modular payoff}}$$

$$\implies \gamma(G)c(b) \geq \sum_{i \in \mathcal{K}(G)} c(e_i^*) + \sum_{i \in \mathcal{K}(G)} |S_i| c(b - e_i^*)$$

$\sigma_i \to \infty$ for every key task i while holding the number of key tasks fixed requires $|S_i| \to \infty$ for every key task i. Then $|S_i| \to \infty \implies e_i^* \to b$ and $\sum_{i \in \mathcal{K}(G)} c(e_i^*) \to \gamma(G)c(b)$. Conclude that $\lim_{|S_i| \to \infty} \sum_{i \in \mathcal{K}(G)} |S_i| c(b - e_i^*) = 0$ by the above inequality and the difference between the modular payoff and efficiency vanishes as $|S_i| \to \infty$. □

For large projects where each key task has many offshoot tasks, the modular assignment is optimal and close to efficient. In the following corollaries, I detail some scenarios where the modular task assignment is always optimal.

Corollary 1. *If the principal needs to complete every task in a key-peripheral project, then the modular assignment is optimal.*

[12] The principal can still uniquely implement the minimum dominating effort profile by assigning an agent to a key task and two linked peripheral tasks, and so the upper bound on the number of assigned agents is $k - 2 \cdot \gamma(G)$ where k is the total number of tasks. Assigning any more agents threatens the uniqueness of implementability.

It is realistic to assume settings where the principal must finish the entire project, and managing team effort costs is a secondary objective. Any assignment that achieves a strictly higher payoff must leave tasks incomplete in equilibrium by Proposition 2, so the modular assignment is optimal.

Corollary 2. *If every key task is linked to strictly more peripheral tasks than key tasks, then the modular assignment is optimal.*

If every key task is linked to more peripheral tasks than key tasks, then $\min_{i \in \mathcal{K}(G)} \sigma_i \geq 1$. The sufficient condition of Theorem 4 holds since $c(b) < 1$ by assumption.

Corollary 3. *The modular assignment is optimal for linear costs and fixed capacity costs of the form:*

$$c(e) = \begin{cases} 0 & 0 \leq e \leq b \\ +\infty & e > b \end{cases}$$

With fixed capacity costs, there are no benefits to dividing tasks among different agents. With linear costs, it is actually harmful to divide tasks within a module among different agents. If an agent assigned to key task i exerts any effort less than b, then the total cost to completing the entire module is at least $c(e_i) + |S_i| c(b - e_i) > c(b)$, which is more expensive than the modular assignment. In either case, the trade-off between free-riding and task over/underload disappears and the optimal assignment must complete every task. The modular assignment does exactly that. The same reasoning applies to arbitrary networks: a generalized modular assignment, which implements a minimum dominating effort profile as a Nash equilibrium by Theorem 2, is optimal for linear and fixed capacity costs.

6.3 When Sufficiency Fails

Part of the appeal of Theorem 4 is that it is agnostic of the functional form of the cost of effort and relies only on computing $c(b)$ and σ_i. The next result shows that finding the optimal assignment on key-peripheral networks with low specialization relies on the shape of the effort cost.

Theorem 5. *If* $\min_{i \in \mathcal{K}(G)} \sigma_i < 1$*, then there exists a cost function that makes the modular assignment optimal and another cost function that makes it suboptimal.*

Proof. See Appendix. □

If a cost function satisfies $c(b) < \min_{i \in \mathcal{K}(G)} \sigma_i$, then the modular assignment is optimal. However, it is not a necessary condition. Constructing a cost function that makes the modular assignment suboptimal requires more care, but the intuition is simple. The principal can exploit the key task with low specialization to reduce task loads and spread effort costs at many neighboring modules. A convex cost function that is close to zero for most of the interval $[0, b]$ suffices as the following example illustrates.

Example 2. The project in Figure 7 has minimum key task specialization of $\frac{1}{2}$. Depicted is an alternate assignment that could beat the modular assignment with the right cost function. It designates the key task with the lowest specialization to support intermediate efforts at neighboring key tasks. Each neighboring key task is assigned to the blue agent and every peripheral task is assigned to a different agent. The peripheral tasks of the top blue key task are left incomplete.

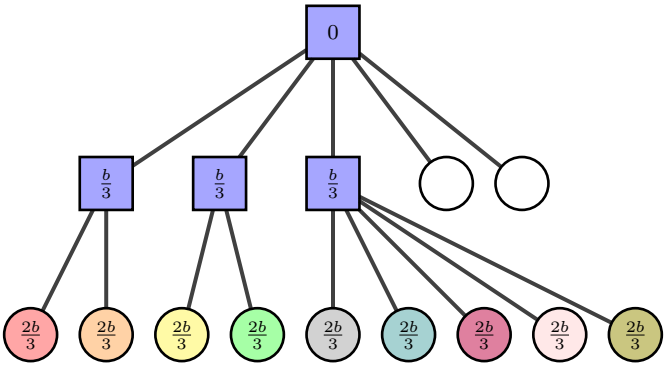


Figure 7: An Alternate Assignment

The total payoff of the implemented equilibrium is $13 - (c(b) + 9 \cdot c(\frac{2b}{3}))$. The payoff from modularization is $15 - 4 \cdot c(b)$, and the alternate assignment does better than the modular assignment if the cost function satisfies $c(b) > \frac{2}{3} + 3 \cdot c(\frac{2b}{3})$. With a completion threshold of $b = 0.99$, the quadratic cost function $c(e) = e^2$ fails the inequality while $c(e) = e^6$ does not. Interestingly, a very convex cost function $c(e) = e^{75}$ fails the inequality yet again because $c(b) = 0.99^{75} \approx 0.471 < \frac{1}{2} = \min_{i \in \mathcal{K}(G)} \sigma_i$ and the modular assignment is optimal.

6.4 Beyond Key-Peripheral Networks

A natural step towards a general characterization of optimal task assignments is to include intermediary tasks that act as helper tasks for multiple key tasks.

Definition 5. An **intermediary** task is linked to at least two key tasks but not any peripheral tasks.

To handle projects with intermediary tasks, I turn to the generalized modular assignment. Recall that it assigns every task in an arbitrary project and implements a minimum dominating effort profile. For projects that contain key, peripheral, and intermediary tasks, a local reduction in the space of assignments comparable to a generalized modular assignment still applies: if an assignment achieves a strictly higher payoff than a generalized modular assignment, then it must implement intermediate effort $e_i \in (0, b)$ at some key task i. For this larger class of project structures, a null result regarding optimality holds.

Theorem 6. *For a project composed of key, peripheral, and intermediary tasks, there exists a cost function that makes a generalized modular assignment optimal and another cost function that makes it suboptimal.*

Proof. See Appendix. □

To make a generalized modular assignment optimal, the desired cost function should ensure spreading costs via convexity is futile. A cost function that is sufficiently close to being linear, such as $c(e) = (\frac{e}{b\beta})^\alpha$ with α sufficiently close to (but greater than) 1 and β sufficiently large does the trick. Concerning suboptimality, the construction of a suitable assignment and cost function is similar to that of Theorem 5 although one has to take extra care in how the intermediary tasks are assigned. The following example illustrates one possible alternate assignment on a connecting star project with an intermediary task. The main takeaway is that outside the class of sufficiently specialized key-peripheral projects, characterizing the optimal assignment requires additional assumptions regarding the effort cost and project structure.

Example 3. The following figure depicts two different assignments on a network with an intermediary task connecting two star graphs.

(a) Generalized Modular Assignment (b) Alternate Assignment

Figure 8: Assignments on a Project with an Intermediary Task

Each color represents a different assigned agent and white tasks remained unassigned. Notice that the presence of the intermediary task can allow all tasks to be completed in an equilibrium implemented by an alternate assignment. The alternate assignment spreads effort across multiple tasks and achieves a higher payoff if $\frac{c(b)}{4} > c(\frac{b}{2})$. Fixing $b = 0.99$, the cubic cost function $c(e) = e^3$ satisfies this inequality but another cost function $c(e) = e^{4/3}$ fails to do so.

7 Extensions

7.1 Imperfect Substitutability

I study the task assignment problem with imperfect substitutability between tasks in this extension. A task i is completed if and only if it is assigned and $e_i + \delta \sum_{\ell \in G_i} e_\ell \geq b$ where δ measures the substitutability of effort between linked tasks. For simplicity, I assume the degree of substitutability is constant across every link. High δ values reflect high levels of effort substitutability where $\delta = 1$ represents perfect substitutability, while low δ values reflect low levels of substitutability. If $\delta = 0$, then links between tasks are superfluous and each task can be treated as a singleton in the project.

A minimum dominating effort profile under imperfect substitutability prescribes effort $\frac{b}{\delta}$ instead of b at tasks in a minimum dominating set. The performance of the modular assignment depends on the value of δ. If δ is very low, then exerting $\frac{b}{\delta}$ at the key task can be expensive for both the principal and the agent. An adequately high δ alleviates this concern.

Theorem 7. *If* $\min_{i \in \mathcal{K}(G)} \sigma_i \geq c(\frac{b}{\delta})$ *and* δ *is sufficiently high (which can depend on the cost function and network structure), the modular assignment implements the minimum dominating set and is optimal for the principal.*

Proof. See Appendix. □

Example 4. I provide intuition for how δ plays a role in determining the implementability and optimality of modularization. Consider the connecting star graph where one key task is linked to three peripheral tasks and the other is connected to two peripheral tasks.

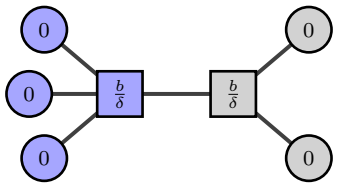


Figure 9: Modular Assignment on the Connecting Star

Implementability: The payoff received by the blue agent is $4 - c(\frac{b}{\delta})$. δ needs to be sufficiently high to prevent the blue agent from having a profitable deviation while the gray agent plays $\frac{b}{\delta}$ at his key task. We can inspect all possible deviations the blue agent has while holding his number of completed tasks constant. If the blue agent seeks to complete only one task within his assigned module, then the cheapest way to do so is by shirking at the key task; δ must satisfy the best response (BR) constraint $1 - c(0) \leq 4 - c(\frac{b}{\delta})$. The least expensive way to complete two tasks within his module is by exerting b at a peripheral task and shirking at the key task; the corresponding constraint is $2 - c(b) \leq 4 - c(\frac{b}{\delta})$. Continuing this pattern, the modular assignment implements the minimum dominating effort profile for the blue agent if δ satisfies the following BR constraints: $4 - c(\frac{b}{\delta}) \geq m - c((m-1)b)$ for $m \in \{1, 2, 3, 4\}$. A δ value that satisfies every BR constraint for the blue agent exists since c is continuous and every inequality holds for $\delta = 1$. Under imperfect substitutability, assigning agents to an entire module has its benefits— a greater gross payoff for agents permits lower values of δ. Ostensibly, a higher δ parameter is needed for more convex cost functions. A computation yields that δ needs to be at least 0.469 to satisfy all BR constraints if $b = \frac{3}{4}$ and a quadratic cost. For a cubic cost and the same completion threshold, δ needs to be greater than 0.558.

For the minimum dominating effort profile to be implementable across the entire project, δ must satisfy the BR constraints for every agent. The sufficient level of δ is higher for the gray agent because he is assigned to a smaller module. The smaller the module, the more expensive it is to exert effort $\frac{b}{\delta}$. Thereby, it is sufficient for δ to satisfy the BR constraints for the agent assigned to the smallest module. Repeating the same

calculations for the gray agent with $b = \frac{3}{4}$, δ must be at least 0.600 for a quadratic cost, while for a cubic cost, the lower bound for δ is 0.667.

Optimality: The crucial insight is that if δ is sufficiently large, then the analysis essentially reduces to that of Theorem 4. Modularization is optimal as long as each key task is sufficiently specialized.

7.2 Understaffing

What kind of task assignments should the principal employ when her team is understaffed? In this extension, I allow there to be fewer agents than tasks in the project i.e. $n < k$. How understaffing affects the principal's optimal assignment depends on how big the project is. If $n \geq \gamma(G)$ where $\gamma(G)$ is the domination number of key-peripheral network G, then the modular assignment is still feasible and Theorem 4 holds. However, if $n < \gamma(G)$, then the comparison across assignments depends on the network structure and costs. Free-riding is less of a problem as task overloading now.

Example 5. If $n = 2$, then using an assignment that assigns the blue and gray agents to some modules completes eight tasks out of ten in the project shown in Figure 10.

(a) Modular Assignment (b) Overloading Modular Assignment

Figure 10: Assignments with $n = 2$

Assignments that spread effort among tasks within modules reap low payoffs when there is understaffing because they require assigning different agents to peripheral tasks. Instead, the principal should utilize an **overloading modular assignment** and assign agents to multiple modules. This accomplishes more tasks but incurs higher costs. If $c(2b) - c(b) \leq 2$, then the gray agent will exert effort level b at both of his key tasks. Moreover, this implies the overloading modular assignment also obtains a higher payoff than the modular assignment.

In the example above, there is only one overloading modular assignment. In larger projects, how modules are divided among agents is central to determining which assignment is best. For instance, in a project with 6 modules and 3 agents, the payoffs to assigning 2 agents to 3 modules each (where each agent exerts effort level $3b$) or 3 agents to 2 modules each (where each agent exerts effort level $2b$) depends on the values of $c(3b)$ and $c(2b)$. As can be seen, the structure of optimal assignments where the team is acutely understaffed i.e. $n < \gamma(G)$ depends on the specific cost function and project.

8 Concluding Remarks

Free-riding is a prevalent issue in teams: agents are prone to shirking at central tasks in the project. This can be mitigated by assigning them to more related tasks, but this can run into the problem of task overload. This paper offers an intuitive solution to both problems via the modular assignment on key-peripheral projects. Assigning an agent to a module of related tasks— a key task and its peripheral tasks— ensures that he will prioritize completing the most important task in his work stream. The modular assignment is optimal on key-peripheral networks where key tasks are sufficiently specialized. Key-peripheral projects with highly specialized key tasks are common and embrace two important principles in systems design: inheritance and loose coupling. Characterizing the optimal task assignment beyond key-peripheral projects requires supplementary details regarding the shape of the cost function and the task structure.

There are multiple directions for future research on the task assignment problem. First, in my model, I take the project structure as a primitive. Adding an initial project planning period that precedes the task assignment phase can capture that, in practice, the principal may have some control over project design. Second, agents may have different skill sets more suitable for some tasks than others. Incorporating heterogeneous productivity among agents across tasks can add another dimension to the trade-off— in addition to mitigating free-riding and task overload, the principal should try to match agents to tasks that fit their skill sets. Third, projects may contain tasks that have to be completed in sequence. The underlying task structure thus takes the form of a directed network, and the optimal task assignment will likely depend on the direction of effort spillovers and substitutability. Lastly, some tasks may take longer than others to complete. Embedding the task assignment problem in a dynamic setting can introduce time-dependent free-riding behavior where agents wait until linked tasks are completed

before starting their task. Equipping the principal with the ability to set deadlines or project milestones in tandem with task assignment power would be interesting to explore.

References

Ahuja, Ravindra K., Thomas L. Magnanti, and James B. Orlin. 1993. Network Flows: Theory, Algorithms, and Applications. Englewood Cliffs, N.J:Prentice Hall.

Allouch, Nizar. 2015. "On the Private Provision of Public Goods on Networks." Journal of Economic Theory, 157: 527–552.

Alonso, Ricardo, and Niko Matouschek. 2008. "Optimal Delegation." The Review of Economic Studies, 75(1): 259–293.

Banks, Jeffrey S., and Rangarajan K. Sundaram. 1998. "Optimal Retention in Agency Problems." Journal of Economic Theory, 82(2): 293–323.

Bramoullé, Yann, and Rachel Kranton. 2007. "Public Goods in Networks." Journal of Economic Theory, 135(1): 478–494.

Bramoullé, Yann, Rachel Kranton, and Martin D'Amours. 2014. "Strategic Interaction and Networks." American Economic Review, 104(3): 898–930.

Chen, Sheu Hua, and Hong Tau Lee. 2007. "Performance Evaluation Model for Project Managers using Managerial Practices." International Journal of Project Management, 25(6): 543–551.

Che, Yeon-Koo, and Seung-Weon Yoo. 2001. "Optimal Incentives for Teams." American Economic Review, 91(3): 525–541.

Chidamber, S.R., and C.F. Kemerer. 1994. "A Metrics Suite for Object Oriented Design." IEEE Transactions on Software Engineering, 20(6): 476–493.

Dasaratha, Krishna, Benjamin Golub, and Anant Shah. 2024. "Incentive Design With Spillovers." Working Paper.

Elliott, Matthew, and Benjamin Golub. 2019. "A Network Approach to Public Goods." Journal of Political Economy, 127(2): 730–776.

Ferguson, J. C., and S. Pruess. 1991. "Shape-preserving interpolation by parametric piecewise cubic polynomials." Computer-Aided Design, 23(7): 498–505.

Fritsch, F. N., and R. E. Carlson. 1980. "Monotone Piecewise Cubic Interpolation."

SIAM Journal on Numerical Analysis, 17(2): 238–246.

Fu, Richard, Ajay Subramanian, and Anand Venkateswaran. 2016. "Project Characteristics, Incentives, and Team Production." Management Science, 62(3): 785–801.

Galeotti, Andrea, Sanjeev Goyal, Matthew O. Jackson, Fernando Vega-Redondo, and Leeat Yariv. 2010. "Network Games." The Review of Economic Studies, 77(1): 218–244.

Garud, Raghu, Richard Langlois, and Arun Kumaraswamy, ed. 2003. Managing in the Modular Age: New Perspectives on Architectures, Networks and Organizations. Oxford:Blackwell.

Georgiadis, George. 2015. "Projects and Team Dynamics." The Review of Economic Studies, 82(1): 187–218.

Georgiadis, George, and Balazs Szentes. 2020. "Optimal Monitoring Design." Econometrica, 88(5): 2075–2107.

Georgiadis, George, Steven A. Lippman, and Christopher S. Tang. 2014. "Project Design with Limited Commitment and Teams." The RAND Journal of Economics, 45(3): 598–623.

Glover, Jonathan, and Eunhee Kim. 2021. "Optimal Team Composition: Diversity to Foster Implicit Team Incentives." Management Science, 67(9): 5800–5820.

Groves, Theodore. 1973. "Incentives in Teams." Econometrica, 41(4): 617–631.

Holmström, Bengt. 1979. "Moral Hazard and Observability." The Bell Journal of Economics, 10(1): 74–91.

Holmström, Bengt. 1982. "Moral Hazard in Teams." The Bell Journal of Economics, 13(2): 324–340.

Holmström, Bengt. 1999. "Managerial Incentive Problems: A Dynamic Perspective." The Review of Economic Studies, 66(1): 169–182.

Holmström, Bengt. 2017. "Pay for Performance and Beyond." The American Economic Review, 107(7): 1753–1777.

Hyväri, Irja. 2006. "Project Management Effectiveness in Project-Oriented Business Organizations." International Journal of Project Management, 24(3): 216–225.

Kinateder, Markus, and Luca Paolo Merlino. 2017. "Public Goods in Endogenous Networks." American Economic Journal: Microeconomics, 9(3): 187–212.

Kinateder, Markus, and Luca Paolo Merlino. 2023. "Free Riding in Networks." European Economic Review, 152: 104378.

Maslach, Christina, and Michael P. Leiter. 2016. "Understanding the Burnout Experience: Recent Research and its Implications for Psychiatry." World Psychiatry, 15(2): 103–111.

Matouschek, Niko, Michael Powell, and Bryony Reich. 2024. "Organizing Modular Production." Working Paper.

Mirrlees, James A. 1976. "The Optimal Structure of Incentives and Authority within an Organization." The Bell Journal of Economics, 7(1): 105–131.

Parnas, D. L. 1972. "On the Criteria to be used in Decomposing Systems into Modules." Communications of the ACM, 15(12): 1053–1058.

Simon, Herbert A. 1965. "The Architecture of Complexity." General Systems : Yearbook of the Society for the Advancement of General Systems Theory, 10: 63.

Stevens, W. P., G. J. Myers, and L. L. Constantine. 1999. "Structured design." IBM Systems Journal, 38(2/3): 231–256.

Subramanyam, R., and M.S. Krishnan. 2003. "Empirical analysis of CK metrics for object-oriented design complexity: implications for software defects." IEEE Transactions on Software Engineering, 29(4): 297–310.

West, Douglas Brent. 2001. Introduction to Graph Theory. Prentice Hall.

A Omitted Proofs

A.1 Proof of Theorem 2

Proof. Consider a project G and denote D_{min} as its minimum dominating set. Assign tasks via the *generalized* modular assignment:

Algorithm 1 Generalized Modular Assignment.

Require: D_{min} a minimum dominating set of graph G.

1: **for** each task $i \in D_{min}$ **do**
2: Assign task i and every unassigned $j \in G_i \setminus D_{min}$ to an unassigned agent.
3: **end for**

I proceed by showing that the corresponding minimum dominating effort profile where agents play full effort b at their task in D_{min} and zero everywhere else is a Nash equilibrium.

Take some agent a assigned to task $i \in D_{min}$. There are two cases to consider. First, assume that i is not linked to any other task in D_{min} i.e. $G_i \cap D_{min} = \emptyset$. If every other agent is playing effort level b at their assigned task in D_{min}, then there are no efforts expended by other agents that contribute to completing task i. Thus it is a best response for agent a to play full effort b at task i. This is because $c(b) < 1$ and doing so completes i along with any other of his assigned tasks in G_j that do not neighbor another task in D_{min}.

Now assume that $G_i \cap D_{min} \neq \emptyset$. By Algorithm 1, agent a must be assigned to an extra task $j \notin D_{min}$ with $G_j \cap D_{min} = \{i\}$. Suppose not, then every neighboring task of i is also linked to some task in $\ell \in D_{min}$ with $\ell \neq i$. Then $D_{min} \setminus \{i\}$ dominates the graph, which contradicts D_{min} being a minimum dominating set. This argument does not depend on the order to which the unassigned tasks $j \in G_i \setminus D_{min}$ are assigned in the for loop. Notice that if every other agent is playing effort level b at their assigned task in D_{min}, then there are no efforts expended by other agents that contribute to completing task j. Thus it is a best response for agent a to play full effort b at task i, which completes task j along with any other task that also does not neighbor another task in D_{min}.

Every task is completed, and the gross principal's payoff is the number of tasks which is k. Recall that $\gamma(G)$ is the size of a minimum dominating set, and so a generalized modular assignment assigns $\gamma(G)$ agents who each exert full effort b in equilibrium for

a total cost of $\gamma(G)c(b)$. $\square$

A.2 Proof of Proposition 1

Proof. The efficient effort profile solves the following cost-minimization problem:

$$\min_{\{e_i\}_{i=1}^k} \sum_{i=1}^{k} c(e_i) \quad \text{s.t. } e_i + \sum_{\ell \in G_i} e_\ell \geq b \text{ and } e_i \geq 0 \text{ for all } i$$

Let $\{e_j^*\}_{j=1}^k$ denote the solution. The first-order conditions of the cost-minimization problem are:

$$c'(e_i^*) = \sum_{\ell \in G_i \cup \{i\}} \lambda_\ell + \mu_i \quad ; \quad c'(e_h^*) = \lambda_i + \lambda_h + \mu_h$$

where λ_ℓ is the Lagrange multiplier on the $e_\ell^* + \sum_{\ell' \in G_\ell} e_{\ell'} \geq b$ constraint and μ_i is the Lagrange multiplier on the nonnegativity constraint. Note that $e_i^* + e_h^* = b$ for all $h \in S_i$, otherwise the total cost is not minimized. Thereby $e_h^* = e_{h'}^*$ for all $h, h' \in S_i$. Before proceeding, I state a helpful Lemma.

Lemma: For all $h \in S_i$, $e_h^* > 0$.

Assume by contradiction that there exists $h \in S_i$ such that $e_h^* = 0$. This implies $e_i^* = b$ and subsequently $e_h^* = 0$ for all $h \in S_i$. Define $L := \{\ell \in G_i : e_\ell^* + \sum_{\ell' \in G_\ell \setminus \{i\}} e_{\ell'}^* = 0\}$. $S_i \subseteq L$ by assumption, so L is non-empty. All of the tasks in L require $e_i^* = b$ so their respective effort thresholds are met. Consider the following ϵ-deviation effort profile: increase effort at all tasks in L by some $\epsilon > 0$ and decrease effort at task i by the same ϵ. This deviating effort profile completes the same number of tasks as before with a lower cost if ϵ is chosen to satisfy $|L| \cdot c(\epsilon) + c(b - \epsilon) < c(b)$.

Since c is C^1 and $c'(0) = 0$ by assumption, there exists $\epsilon_0 > 0$ such that for all $\epsilon_0' < \epsilon_0$, $\frac{c(\epsilon_0')}{\epsilon_0'} < \frac{c'(0.99b)}{|L|}$. There also exists $\epsilon_1 > 0$ such that for all $\epsilon_1' < \epsilon_1$, $\frac{c(b) - c(b - \epsilon_1')}{\epsilon_1'} > c'(0.99b)$ because c is strictly increasing and convex. Set $\epsilon = \min\{\epsilon_0, \epsilon_1\}$ so that $\frac{|L| \cdot c(\epsilon)}{\epsilon} < \frac{c(b) - c(b - \epsilon)}{\epsilon}$ which implies $|L| \cdot c(\epsilon) + c(b - \epsilon) < c(b)$ and proves the Lemma.

By the Lemma, $e_h^* > 0$ and so $\mu_h = 0$. Moreover, $e_i^* + |S_i| \cdot e_h^* > b$, which implies $\lambda_i = 0$. The first order condition can be simplified:

$$c'(e_h^*) = \lambda_i + \lambda_h + \mu_h \implies c'(e_h^*) = \lambda_h$$

$$c'(e_i^*) = \lambda_i + \sum_{h \in S_i} \lambda_h + \sum_{j \in G_i \setminus S_i} \lambda_j + \mu_i = |S_i| \cdot c'(e_h^*) + \sum_{j \in G_i \setminus S_i} \lambda_j$$

Since all other neighboring task nodes $j \in G_i \setminus S_i$ are also key tasks, then $\lambda_j = 0$ for all $j \in G_i \setminus S_i$. Then $\sum_{j \in G_i \setminus S_i} \lambda_j = 0$ and $c'(e_i^*) = |S_i| \cdot c'(e_h^*) = |S_i| \cdot c'(b - e_i^*)$. □

A.3 Proof of Theorem 5

Proof. Fix some network G with minimum specialization $\min_{i \in \mathcal{K}(G)} \sigma_i < 1$. By Theorem 4, the modular assignment is optimal for any cost function with $c(b) \leq \min_{i \in \mathcal{K}(G)} \sigma_i$.

Now I construct an alternate assignment and a suitable cost function that makes the modular assignment suboptimal. A cost function satisfying $c(b) > \min_{i \in \mathcal{K}(G)} \sigma_i$ is not enough as Example 2 demonstrates; the shape of the cost function matters. Label the key task with lowest specialization as i and consider an alternate task assignment that uses i to support intermediate efforts at every neighboring key task: assign all tasks in $\{i\} \cup (G_i \setminus S_i)$ to one agent (call him agent a), each peripheral task $k \in S_j$ for key task $j \in G_i \setminus S_i$ to a different agent, and all other modules following the modular assignment. This assignment keeps the tasks in S_i unassigned.

Now consider the effort profile where agent a plays effort $\frac{b}{|G_i \setminus S_i|}$ at each task in $G_i \setminus S_i$ and zero effort at task i. Each agent assigned to a peripheral task of a key task residing in $G_i \setminus S_i$ plays effort $\frac{b(|G_i \setminus S_i| - 1)}{|G_i \setminus S_i|}$. Every other agent assigned to a module plays effort b at his respective key task and zero at his peripheral tasks.

Agent a does not have a profitable deviation in lowering his effort at any of his tasks lest he loses task i's completion payoff. Every agent assigned to a peripheral task of a key task in $G_i \setminus S_i$ is playing a best response. The alternate assignment implements said effort profile as a Nash equilibrium with intermediate effort at key tasks in $\{i\} \cup (G_i \setminus S_i)$ and replicates the modular payoff everywhere else. Thus, it is without loss to compare the payoffs between the alternate assignment and the modular assignment on the key tasks residing in $\{i\} \cup (G_i \setminus S_i)$ and their peripheral tasks. The payoff to the alternate task assignment is:

$$\underbrace{\sum_{j \in G_i \setminus S_i} |S_j| + |\{i\} \cup (G_i \setminus S_i)|}_{\text{total \# of tasks}} - \underbrace{c\left(0 + (|G_i \setminus S_i|) \cdot \tfrac{b}{|G_i \setminus S_i|}\right)}_{= c(b) \text{ ; agent } a\text{'s cost}} - \sum_{j \in G_i \setminus S_i} |S_j| \cdot \underbrace{c\left(\tfrac{b(|G_i \setminus S_i| - 1)}{|G_i \setminus S_i|}\right)}_{\text{each peripheral agent's cost}}$$

The payoff to the modular assignment is:

$$|S_i| + \sum_{j \in G_i \setminus S_i} |S_j| + |\{i\} \cup (G_i \setminus S_i)| - |\{i\} \cup (G_i \setminus S_i)| \cdot c(b)$$

The alternate assignment achieves a strictly higher payoff if:

$$|G_i \setminus S_i| \cdot c(b) > |S_i| + \sum_{j \in G_i \setminus S_i} |S_j| \cdot c(\tfrac{b(|G_i \setminus S_i|-1)}{|G_i \setminus S_i|})$$

The goal is to construct a cost function $c : \mathbb{R}^+ \to \mathbb{R}^+$ that is C^1, strictly increasing and strictly convex with $c(0) = 0$ and $c'(0) = 0$ that satisfies the above inequality. Set $c(b) = \frac{1}{2}(\frac{|S_i|}{|G_i \setminus S_i|} + 1)$. By assumption, $\min_i \sigma_i = \frac{|S_i|}{|G_i \setminus S_i|} < 1$, so $\min_i \sigma_i < c(b) < 1$ and there exists some $\epsilon \in (0, 1)$ such that $c(b) = \epsilon + \frac{|S_i|}{|G_i \setminus S_i|}$. Set $c(\frac{b(|G_i \setminus S_i|-1)}{|G_i \setminus S_i|}) = \frac{\epsilon \cdot |G_i \setminus S_i|}{2m \sum_{j \in G_i \setminus S_i} |S_j|}$ where m is a natural number chosen such that $\frac{\epsilon \cdot |G_i \setminus S_i|}{2m \sum_{j \in G_i \setminus S_i} |S_j|} < \frac{1}{2}(\frac{|S_i|}{|G_i \setminus S_i|} + 1)(\frac{(|G_i \setminus S_i|-1)}{|G_i \setminus S_i|})$, where the latter term is the secant line between $(0, 0)$ and $(b, \frac{1}{2}(\frac{|S_i|}{|G_i \setminus S_i|} + 1))$ evaluated at $\frac{b(|G_i \setminus S_i|-1)}{|G_i \setminus S_i|}$. This ensures convexity is satisfied and the alternate assignment achieves a strictly higher payoff than the modular assignment. We now have three points necessary to construct a polynomial: $(0, 0)$, $(\frac{b(|G_i \setminus S_i|-1)}{|G_i \setminus S_i|}, \frac{\epsilon \cdot |G_i \setminus S_i|}{2m \sum_{j \in G_i \setminus S_i} |S_j|})$ and $(b, \frac{1}{2}(\frac{|S_i|}{|G_i \setminus S_i|} + 1))$. The desired C^1, strictly monotone and convex cost function with a specified endpoint derivative $c'(0) = 0$ can be generated by the Fritsch-Carlson method for shape-preserving piecewise cubic Hermite polynomial interpolation (see Fritsch and Carlson (1980) and Ferguson and Pruess (1991)). □

A.4 Proof of Theorem 6

Proof. Let G be a network composed of key, peripheral, and intermediary tasks.

Cost Function for Optimality: Any assignment that achieves a strictly higher payoff than the modular must implement an equilibrium with intermediate effort $e_i \in (0, b)$ at some key task i and complete all of its peripheral tasks i.e. efforts $b - e_i$ played. For this to be an equilibrium, the agent assigned to key task i, call him agent a, must be assigned to a supporting neighboring task j. This supporting task can be either key or intermediary.

Case #1: Task j is key. Then its peripheral tasks must be incomplete, and the modular assignment does better if $c(b) < \min_{i \in \mathcal{K}(G)} \sigma_i$. An example is cost function $c(e) = \left(\frac{e}{\beta b}\right)^\alpha$ for some fixed $\alpha > 1$ with β sufficiently large.

Case #2: Task j is intermediary. First note the efficient key task effort in a project that contains intermediary tasks is the same as that of a standard key-peripheral project i.e. $c'(e_i^*) = |S_i|c'(b - e_i^*)$. This can be shown by solving a similar cost-minimization problem to that of Proposition 1. Now consider the class of cost functions $c(e) = \left(\frac{e}{\beta b}\right)^\alpha$ with $\alpha > 1$. Then the efficient key task effort is:

$$e_i^* = b\left(\frac{\exp\left(\frac{\ln(|S_i|)}{\alpha-1}\right)}{1+\exp\left(\frac{\ln(|S_i|)}{\alpha-1}\right)}\right)$$

By L'Hôpital's rule, $\lim_{\alpha\searrow 1} e_i^* = b$, and so as $\alpha \searrow 1$, the generalized modular assignment payoff converges to the efficient payoff.

Now returning to the original inquiry, if task j is intermediary, then there must be at least one key task in $k \in G_j$ different from i by definition. In an equilibrium where task j supports an intermediate effort at task i, it must be that $e_j + \sum_{\ell\in G_j} e_\ell = b$ or else agent a has a profitable deviation by reducing his effort at task i. The upper bound to the payoff of the alternate assignment on tasks in $\{i, j, k\} \cup S_i \cup S_k$ is to complete all tasks at a minimum cost according to:

$$\min_{e_i,e_j,e_k} c(e_i) + |S_i|c(b - e_i) + c(e_j) + c(e_k) + |S_k|c(b - e_k)$$

$$\text{s.t. } e_j + \sum_{\ell\in G_j} e_\ell = b \; ; e_i, e_j, e_k \geq 0$$

Denote $\hat{e}_i$ and $\hat{e}_k$ to be the solution key task efforts at key tasks i and k. For the class of cost functions $c(e) = \left(\frac{e}{\beta b}\right)^\alpha$ with $\alpha > 1$, it cannot be that both $\lim_{\alpha\searrow 1} \hat{e}_i = b$ and $\lim_{\alpha\searrow 1} \hat{e}_k = b$ or else the constraint $e_j + \sum_{\ell\in G_j} e_\ell = b$ is violated. This implies that the payoff to an assignment that implements an equilibrium with intermediate effort $e_i \in (0, b)$ and utilizes a supporting intermediary task j does not converge to efficiency as $\alpha \searrow 1$. Thus for some α, the generalized modular assignment payoff is greater.

Now to tie both cases together, the modular assignment is optimal for cost function $c(e) = \left(\frac{e}{\beta b}\right)^\alpha$ for some α and sufficiently large β.

Cost Function for Suboptimality: Let $\mathcal{I}(G)$ be the set of all intermediary tasks of project G. Choose some intermediary task j such that $G_j \cap D_{min} \not\supset G_k \cap D_{min}$ for all $k \in \mathcal{I}(G) \cap G_j$. Assign intermediary task j and all key tasks $G_j \cap D_{min}$ to some agent, call him agent a. Assign every peripheral task of every key task in $G_j \cap D_{min}$ to a different

unassigned agent. Define the set $\mathcal{J} := \{k \mid G_k \cap D_{min} \subseteq (G_j \cap D_{min})\}$ and notice that $\mathcal{J} \cap G_j = \emptyset$ by property of task j. Assign every task in $\mathcal{J}$ to a different unassigned agent. For all remaining tasks $\ell \in D_{min}$, assign ℓ and its neighboring tasks according to a generalized modular assignment as per Algorithm 1 to different unassigned agents.

This assignment implements the following effort profile as a Nash equilibrium. Agent a exerts effort $\frac{b}{|G_j|}$ at every task in G_j and zero effort at task j. An agent assigned to a peripheral task that neighbors a key task in $G_j \cap D_{min}$ best responds by playing effort $\frac{b(|G_j|-1)}{|G_j|}$. An agent assigned to a task $k \in \mathcal{J}$ with $G_k \cap \mathcal{J} = \emptyset$ and x neighbors in $G_j \cap D_{min}$ best responds with effort $\frac{b(|G_j|-x)}{|G_j|}$. Every other agent assigned to a task in $\mathcal{J}$ plays zero effort. All other agents play effort b at their assigned task in D_{min} and zero elsewhere.

The nontrivial best response to check is that of agent a. If he shirks at any of his tasks in G_j, then he loses the task completion payoff at task j. $\mathcal{J} \cap G_j = \emptyset$ ensures that any of the positive efforts played at intermediary tasks in $\mathcal{J}$ do not contribute to the total efforts at task j. Playing effort b at task j and zero at every other task in G_j is not a profitable deviation because every task in G_j is completed regardless of his effort provision. This is because the tasks in $G_j \cap D_{min}$ neighbor at least two peripheral tasks with efforts $\frac{b(|G_j|-1)}{|G_j|}$ with $|G_j| \geq 2$, so the total neighboring efforts total to at least the completion threshold b.

The payoff to this assignment replicates a generalized modular assignment everywhere except on the tasks in $\{j\} \cup G_j \cup \mathcal{J}$. It is easy to see that the completion threshold is met at tasks in $G_j \cup \{j\}$ and the assigned tasks in $\mathcal{J}$. Thus the alternate assignment completes every task in $\{j\} \cup G_j \cup \mathcal{J}$ with an upper bound cost of:

$$c(|G_j| \cdot \tfrac{b}{|G_j|}) + |\mathcal{J}| \cdot c(\tfrac{b(|G_j|-1)}{|G_j|})$$

A generalized modular assignment counts a cost of $|G_j \cap D_{min}| \cdot c(b)$ on the tasks in $\{j\} \cup G_j \cup \mathcal{J}$. Thus the alternate assignment achieves a strictly higher payoff if:

$$|\mathcal{J}| \cdot c(\tfrac{b(|G_j|-1)}{|G_j|}) < (|G_j \cap D_{min}| - 1)c(b) \iff c(\tfrac{b(|G_j|-1)}{|G_j|}) < \tfrac{|G_j \cap D_{min}|-1}{|\mathcal{J}|}c(b)$$

Now to construct the desired cost function, set $c(b)$ to some value $C \in (0, 1)$ and set $c(\frac{b(|G_j|-1)}{|G_j|}) = \frac{|G_j \cap D_{min}|-1}{2m|\mathcal{J}|}$ where m is a natural number chosen such that $\frac{|G_j \cap D_{min}|-1}{2m|\mathcal{J}|} < C \cdot (\frac{b(|G_j|-1)}{|G_j|})$ to preserve convexity. Again, we appeal to the Fritsch-Carlson method for shape-preserving piecewise cubic Hermite polynomial interpolation to generate the

desired cost function (see Fritsch and Carlson (1980) and Ferguson and Pruess (1991)). □

A.5 Proof of Theorem 7

Proof. The proof proceeds in two parts: showing that the modular assignment implements the minimum dominating effort profile under imperfect substitutability for a sufficiently large δ and that the modular assignment is optimal when $c(b) \leq \min_{i \in \mathcal{K}(G)} \sigma_i$ for a sufficiently large δ.

Implementability: The modular assignment assigns each agent to a module; an agent assigned to module $S_i \cup \{i\}$ receives payoff of $|S_i| + 1 - c(\frac{b}{\delta})$ if he plays according to the equilibrium effort profile.

We can inspect all possible deviations the agent has by holding his number of completed tasks in the module constant, which creates $|S_i|$ separate cost minimization problems and generates $|S_i|$ best-response (BR) constraints. We consider deviations that set $e_i \neq \frac{b}{\delta}$, lest they would not be profitable. The agent's best payoff from completing m tasks within the module $S_i \cup \{i\}$ with $e_i \neq \frac{b}{\delta}$ solves:

$$\min e_i + \sum_{\ell \in S_i} e_\ell$$

$$\text{s.t. } e_h + \delta \sum_{\ell \in G_h} e_\ell \geq b \text{ for exactly } m \text{ tasks and } e_h \geq 0 \text{ for all tasks } h \in S_i \cup \{i\}$$

Since all other agents are playing according to the minimum dominating effort profile, key task i can be completed by free-riding. The solution to the problem above for $m = 1$ is setting $e_i = e_l = 0$ for all $l \in S_i$, for a payoff of 1. To complete 2 tasks, the solution is to set one of the $e_l = b$ and the rest of the efforts equal to zero, for a payoff of $2 - c(b)$. To complete 3 tasks, the solution is to set $e_l = b$ for 2 tasks for a payoff of $3 - c(2b)$. Thereby, for playing $\frac{b}{\delta}$ at key task i to be a best response, the following constraints must hold:

$$|S_i| - c(\tfrac{b}{\delta}) \geq m - c((m-1)b) \text{ for all } m \in \{1, 2, ..., |S_i|\}$$

A sufficiently high δ that is less than 1 that satisfies all BR constraints exists because c is continuous and $|S_i| \geq 2$. Take the maximum over all δ values that make each agent's best response $e_i = \frac{b}{\delta}$ at their key task. This is the value of δ that makes the modular assignment implement the minimum dominating effort profile under imperfect

substitutability.

Optimality: The payoff from the modular assignment is:

$$\sum_{i \in \mathcal{K}(G)} (|S_i| + 1) - |\mathcal{K}(G)| \cdot c(\tfrac{b}{\delta})$$

We break the analysis into cases for an arbitrary module with key task i with possible deviate efforts of $e_i < \frac{b}{\delta}$:

Case #1: Principal implements effort $e_i = 0$. Then the greatest payoff possible is if the principal assigns all other peripheral tasks to different agents. For modularization to be optimal on module $S_i \cup \{i\}$, the following inequality must hold:

$$|S_i| + 1 - c(b/\delta) \geq |S_i| + 1 - |S_i| \cdot c(b) \implies c(\tfrac{b}{\delta}) \leq |S_i| \cdot c(b)$$

Since the inequality holds when $\delta = 1$ and c is continuous, then there must be some $\delta_1 < 1$ such that the inequality holds for all $\delta > \delta_1$.

Case #2: Another deviation is to implement effort $e_i \in (0, \frac{b}{\delta})$. Split into further cases by varying the number of peripheral tasks that are assigned.

Case #2.1: If all of the peripheral task nodes remain unassigned, then obviously none of them are completed and this is sub-optimal compared to the modular assignment.

Case #2.2: Assign $m \in \{1, ..., |S_i| - 1\}$ peripheral tasks to different agents. Doing so beats any principal payoff from assigning any of the same tasks to one agent by convex cost superadditivity. Each agent exerts effort level $e_h = b - \delta e_i$. The maximum payoff of $m + 1 - c(e_i) - m \cdot c(e_h)$ which is bounded from above by $m + 1$, which itself is bounded above by $|S_i|$. Then there exists δ_2 such that $|S_i| + 1 - c(\frac{b}{\delta_2}) > |S_i|$ again by continuity of c.

Case #2.3: Assign all tasks in S_i to different agents who each exert $e_h = b - \delta e_i$. The total effort level contributing to task i is:

$$e_i + \delta |S_i|(b - \delta e_i) + \delta \sum_{\ell \in G_i \setminus S_i} e_\ell$$

Observe that for any $e_i \in (0, \frac{b}{\delta})$:

$$\delta > \frac{1}{|S_i|} \implies e_i + \delta |S_i|(b - \delta e_i) > b$$

Set $\delta_4 > \frac{1}{|S_i|}$. Therefore implementing an intermediate effort level and completing every task within the module is by itself not an equilibrium.

Label $\delta_i = \max\{\delta_1, \delta_2, \delta_3, \delta_4\}$ for module with key task i. Now take $\delta = \max_{i \in \mathcal{K}(G)} \delta_i$. The remainder of the proof of optimality now mirrors that of Theorem 4. Implement a key task effort of $e_i \in (0, \frac{b}{\delta})$ and assign all tasks in S_i to different agents. Since δ is sufficiently high, the principal needs to give the agent assigned to key task i a neighboring key task j that supports it in equilibrium. Just like in Proposition 2, the sum of total efforts at key task j must be exactly b i.e. $e_j + \delta \sum_{\ell \in G_j} e_\ell = b$ for this to be an equilibrium. If the principal assigns one of the peripheral tasks ℓ of key task j to an agent, then he will best respond by playing effort $e_\ell = b - \delta e_j$. Note that there exists a sufficiently large δ such that $e_j + \delta e_i + \delta(b - \delta e_j) > b$ since the inequality holds with $\delta = 1$ and the expression is continuous in δ. Thereby if δ is large enough, then none of the peripheral tasks of a supporting key task can be completed in equilibrium. Then all of the conclusions in Theorem 4 follow in the same fashion and modularization is optimal if $c(\frac{b}{\delta}) \leq \min_{i \in \mathcal{K}(G)} \sigma_i$. □