HARVARD | BUSINESS | SCHOOL

Field Evidence on Individual Behavior & Performance in Rank-Order Tournaments

Kevin J. Boudreau
Constance E. Helfat
Karim R. Lakhani
Michael Menietti

Working Paper

13-016

August 9, 2012

FIELD EVIDENCE ON INDIVIDUAL BEHAVIOR & PERFORMANCE IN RANK-ORDER TOURNAMENTS*

Kevin J. Boudreau,
London Business School, Harvard Business School, Harvard-NASA Tournament Laboratory
kboudreau@london.edu
Constance E. Helfat,
Constance.E.Helfat@tuck.dartmouth.edu
Tuck School of Business at Dartmouth College
Karim R. Lakhani,
k@hbs.edu
Harvard Business School, Harvard-NASA Tournament Laboratory
Michael Menietti
mmenietti@fas.harvard.edu
Harvard University, Harvard-NASA Tournament Laboratory

August 9, 2012

Abstract

Economic analysis of rank-order tournaments has shown that intensified competition leads to declining performance. Empirical research demonstrates that individuals in tournament-type contests perform less well on average in the presence of larger number of competitors in total and superstars. Particularly in field settings, studies often lack direct evidence about the underlying mechanisms, such as the amount of effort, that might account for these results. Here we exploit a novel dataset on algorithmic programming contests that contains data on individual effort, risk taking, and cognitive errors that may underlie tournament performance outcomes. We find that competitors on average react negatively to an increase in the total number of competitors, and react more negatively to an increase in the number of superstars than non-superstars. We also find that the most negative reactions come from a particular subgroup of competitors: those that are highly skilled, but whose abilities put them near to the top of the ability distribution. For these competitors, we find no evidence that the decline in performance outcomes stems from reduced effort or increased risk taking. Instead, errors in logic lead to a decline in performance, which suggests a cognitive explanation for the negative response to increased competition. We also find that a small group of competitors, who are at the very top of the ability distribution (non-superstars), react positively to increased competition from superstars. For them, we find some evidence of increased effort and no increase in errors of logic, consistent with both economic and psychological explanations.
JEL Codes: D03

*Jack Hughes, Robert Hughes, Ira Heffan, and Mike Lydon from TopCoder generously provided their time and assistance with this paper. Seminar participants at Duke University and Harvard Business School provided feedback on this paper. The Harvard-NASA Tournament Laboratory supported this work. All mistakes remain our own.

1 Introduction

Rank-order tournaments and contests have attracted great interest since the seminal work of Lazear and Rosen (1981) on optimal labor contracts. Tournament theory has been applied to a range of diverse activities, including academic achievement, amateur and professional sports, arts, architecture, manual labor, sales, engineering and scientific work, and executive promotion.[1] Prize-based contests have also been studied as a means to spur innovation (Wright, 1983; Kremer, 1998; Scotchmer, 2004; Terwiesch and Xu, 2008; Kremer and Williams, 2010). These types of contests played an important historical role in driving technological development in a range of industries including agriculture (Brunt, Lerner, and Nicholas, 2011). More recently, large-scale online contest platforms that provide on-going tournament-based work and compensation have emerged in areas such as scientific problem-solving, software development, graphic arts design and creative performance, as anticipated by Autor (2001). Today large industrial companies increasingly use online contests as a complement to in-house research and development.

Theories of behavior and performance in rank-order tournaments have analyzed the provision of economic incentives, as well as strategic responses on the part of participants when contests are used to elicit effort (Lazear and Rosen, 1981; Taylor, 1995; Prendergast, 1999; Moldovanu and Sela, 2001). One issue in the design of tournaments concerns the effect of the number and skill distribution of competitors on the elicitation of effort and performance outcomes. Empirical research on tournaments has shown that individuals perform less well on average when faced with a larger number of competitors in total and a larger number of superstars. This evidence has come primarily from experimental settings and sporting events. Particularly in field settings, studies often lack direct evidence about the underlying mechanisms, such as the amount of effort, that might account for these results. In this study, we exploit a novel dataset on algorithmic programming contests that contains data on individual effort, risk taking, and cognitive errors, enabling us to shed greater light on the mechanisms that underlie performance outcomes in the face of increased competition.

As a simple example of a one-shot tournament, consider amateur golfers playing a round. Players with similar handicaps, i.e., of equal ability, might prefer to play in a twosome rather than a foursome, because the probability of out-playing a single competitor exceeds that of out-playing three rivals. Additionally, each golfer may exert less effort and therefore perform less well in the foursome, because the lower likelihood of winning reduces the returns to effort. Empirical studies of tournament-type contests have shown that, on average, individuals perform less well when faced with a larger number of competitors, including in retail sales (Casas-Arce and Martínez-Jerez, 2009), research tournaments (Fullerton et al., 2002), and software development (Boudreau, Lacetera, and Lakhani, 2011).

Early models of tournaments relied on the assumption that agents had equal ability (Lazear and Rosen, 1981). Subsequent theoretical analysis and empirical research has examined tournaments in which competitors have heterogeneous abilities (Knoeber and Thurman, 1994), including recent work on the effect of completely dominant competitors (Brown, 2011).[2] Continuing with the golf analogy, suppose that Tiger Woods replaces one of the members of the foursome-an extreme form of varying the ability of competitors. On his worst day, Tiger Woods will outplay each of the other golfers. In this extreme case, a superstar competitor performs better with zero effort than another competitor performs with maximum effort; the expected rank of performance for each player therefore drops with virtual certainty relative to contests without such a superstar. If contest payoffs are winner-take-all or strongly non-linear with respect to rank, non-superstars have little incentive to exert effort. Evidence of a ``superstar effect'', in which non-superstars perform less well in the presence of a much superior player, has been found in competitive sporting environments such as tennis (Sunde, 2003; Lallemand, Plasman, and Rycx, 2008), and indeed in the case of professional golf in the US (Brown, 2011) and Japan (Tanaka and Ishino, 2012).[3]

[1] Konrad (2009) provides a comprehensive literature review in a range of contest and tournament settings.

[2] Szymanski and Valletti (2005) consider the implications of tournaments with unequal ability distributions. Brown (2011) provides a simple formal model of the effects of incentives and strategies created by the presence of a superstar in a professional sports setting.

[3] Competitive rank order tournaments may have other disadvantages. Relative to more conventional contracting with a single agent and rewards based on observable outcomes (Holmstrom and Milgrom, 1991), tournaments create redundant costs and efforts by multiple agents on the same task (Che and Gale, 1983; Fullerton and McAfee, 1999). However, by enabling comparisons through relative performance evaluation, tournaments have the advantage of generating more information regarding things such as the difficulty of the task and the relative skills or efforts of workers in the tournament. And, adding at least a minimum level of competition can stimulate effort by discouraging slack (Taylor, 1995). A large number and

Prior research has often attributed the superstar effect to a rational decrease in effort by non-superstars. But in addition, a superstar might affect the behavior and performance of competitors through other channels. Consider this quote from Riley (2012):

> ``There were a number of years, enough to be called an unmatchable era, when Tiger [Woods] won every single tournament in which he held a third-round lead. Part of this dominance was aided by the thumb-sucking meltdowns of his playing partners; when paired with Tiger, opponents put up notably worse scores than they did with anyone else. And these were great players, major-championship contenders. In the past, guys like Robert Rock would stub themselves right out of contention, their intestines were knotted so tightly.

This passage provides a complementary view of the behavioral mechanisms set in motion by intensified competition and the associated impact on performance. Riley interprets the reaction of other players to Tiger Woods as a form of ``choking,'' in which psychological pressure, in this case from the presence of a dominant competitor, causes other golfers to perform below their abilities. Research in psychology has established that stress or pressure on individuals to perform well--including time pressure to perform a task, high stakes, the presence of an audience, and social anxiety--causes individuals to perform below their abilities (Baumeister and Showers, 1986; DeCaro et al., 2011).[4]

Psychological response to competition may take other forms as well. For example, tournament incentives may lead individuals to lower their ``cognitive effort''--also known as mental effort, denoting intensity of mental activity including the degree of voluntary attention or concentration (Kahneman, 1973)--which contrasts with physical or ``labor effort'' (Bracha and Fershtman, 2012). For example, in an experimental study, Bracha and Fershtman (2012) find that some individuals may work harder in terms of labor effort in a tournament-type contest than under a pay-for-performance reward structure, but cognitive effort may deteriorate, causing worse performance outcomes.

Yet other research has suggested that the psychological response to pressure may cause the performance of some competitors to improve rather than decline, particularly for high-ability individuals. In particular, self-confidence in the task at hand, which may be positively correlated with ability, can counterbalance the debilitating effects of stress in some circumstances (Baumeister and Showers, 1986). In an experimental study involving basketball free throws, Otten (2009) finds that greater ``perceived control'' (self-confidence) has a positive affect on performance under pressure, suggesting a possible explanation for ``clutch'' (better than usual) performance.

In addition to the foregoing explanations of reactions to increased competition, both economic and psychological research suggests that the ability of competitors may affect their reactions. For example, economic logic suggests that lower skilled competitors may react less negatively to the presence of a superstar, because a superstar has less impact on the likely rank of these competitors (Brown, 2011). In contrast, high ability competitors on the edge of winning positions may even increase their effort, because they have the most to gain (Casas-Arce and Martínez-Jerez, 2009). Psychological research also suggests that the strongest reactions may come from higher ability competitors, because high performance expectations for these competitors, both from others and from the competitors themselves, may create the greatest pressure to perform well (Baumeister and Showers, 1986).

Taken as a whole, prior research suggests that in addition to providing evidence regarding the underlying mechanisms that may account for performance outcomes in the face of increased competition, it is useful to examine which competitors react most strongly to increased competition, who they react most strongly to, and whether they react positively or negatively. We analyze these issues using a detailed microeconomic data set from tournament type contests sponsored by an online platform that produces commercial grade enterprise software and algorithmic analytics solutions.[5] We obtained data on over 4000 computer programmers who participated in con-

diverse set of competitors may be especially important in cases in which it is not known ex ante which individual will turn out to be best suited to perform a task (Boudreau, Lacetera, and Lakhani, 2011).

[4] A review of the neurobiology literature by Arnsten (2012) covers several recent findings on the interaction of stress and higher brain functions, including the role of stress hormones in causing ``a rapid and dramatic loss of prefrontal cognitive abilities,'' thus impacting an individual's working memory, the short-term memory used during computation.

[5] Boudreau, Lacetera, and Lakhani (2011) use the same data set to examine the relationship between an increasing number of contestants and negative incentive effects as well as extreme-value outcomes from the point of view of a contest sponsor. They do not examine individual responses or the role of superstars.

tests requiring the creation of software solutions to three algorithmic problems at a time. The data have several desirable features, including random assignment of competitors to virtual rooms in which they compete, multiple observations per individual, a fine-grained measure of ability for individual competitors, and performance outcomes per individual in each contest (namely, a problem-solving score and an indicator of the presence of logical errors). The data structure allows us to estimate the impact of the number and skill distribution of competitors on individual performance, while controlling for programming ability and other individual- and contest-specific effects. We also can examine how these effects on individual performance vary by competitor ability. Additionally, the data contain information about the choices and actions of individual competitors, such as the amount of time spent working on a problem. These data provide evidence regarding underlying mechanisms--such as effort, risk taking, and cognitive errors--that may account for performance outcomes in the face of increased competition.

We begin by documenting performance outcomes in response to increased competition. As part of this analysis, we bring together two previously separate research streams: one that has focused on the impact of the total number of competitors on performance outcomes, and another that has focused on the impact of superstar competitors. We first seek to replicate the result that contest participants have worse performance outcomes on average when faced with a larger total number of competitors, or the ``N-effect'' for short.[6] Consistent with prior research, our results show that a larger number of competitors leads to worse performance outcomes on average. We then add the presence of superstars to the analysis, and find that this has an additional negative effect on performance outcomes. Notably, both effects hold in the same setting and data. We then decompose the total number of competitors into superstars and non-superstars, and find that on average participants react much more negatively to an increase in the number of superstars than to an increase in the number of non-superstars. We also find that these negative effects are strongest in a particular subgroup of competitors: those that are highly skilled, but whose abilities put them near to the top rather than at the top of the ability distribution. In addition, we find that a small group of competitors, who are at the very top of the ability distribution (excluding superstars), react positively to increased competition from superstars in particular.

Then we turn to an exploration of the underlying causes of these effects, particularly in higher ability competitors that react most strongly to increased competition. For the near-to-the-top competitors that react most negatively, we find no evidence that increased competition affects observable actions and strategic choices. For example, an increase in the number of superstar and non-superstar competitors does not lead to either lower observable effort or greater risk-taking. Instead, these competitors make more errors of logic in response to greater numbers of superstars and non-superstars, which points to a cognitive explanation for the decline in performance. In addition, for the small group of competitors at the very top of the ability distribution, we find some evidence of increased effort and no increase in errors of logic, consistent with both an economic argument that competitors on the edge of winning may exert maximum effort and psychological theories of self-confidence under pressure.

The paper proceeds as follows. section 2 describes the empirical context in detail. section 3 describes the data set, variables, and estimation approach. section 4 through section 7 present results and analysis. section 8 concludes.

2 Empirical Context: Algorithm Contests at TopCoder Inc.

Data for our study comes from TopCoder, Inc., a web-based platform that delivers outsourced software solutions for its clients through the use of online rank-order tournaments involving a member base of over 400,000 registered software developers (often referred to as ``coders''). Established in 2001, TopCoder works with large information technology intensive organizations (e.g., United Technologies, UBS, ING, IMS Health) to identify their software requirements, which it converts into contests for its member base. These contests are open to all software developers registered on the site and last several days or weeks; prize pools vary from $500 to more than $50,000 depending on the contest. Since its founding, the firm has transferred more than $35 million in prize money and peer review fees to its members by conducting more than 10,000 tournaments for over 200 clients.

[6] This term was coined by Garcia and Tor (2009) in a study of students taking an SAT test in the presence of greater numbers of test-takers. We use the term in the context of tournaments.

TopCoder's value proposition to its clients is the availability of a highly talented pool of software developers that are interested in competing to provide client software solutions. The clearest signal of TopCoder's ability to deliver on the availability of talented software developers is the number of members that have received a skills rating, which measures programming ability. Currently almost 50,000 members have received a TopCoder skills rating, the vast majority through a series of ongoing algorithmic contests called single round match (SRM), which occur online on a weekly to bi-weekly basis. Beyond providing a signal of platform capability to corporate clients, these contests serve to recruit members by creating a tournament environment where developers can demonstrate their skills against a global pool of competitors.

In this study, we analyze participation and performance data from these algorithm contests. TopCoder gave us access to the algorithm contest data (2001-2007), and we conducted extensive interviews with the firm's executive team, its clients, and 20 elite members to improve our understanding of the contest setting.

2.1 Contest Structure

The main task in an algorithm contest is to solve three computer science algorithm problems in 75 minutes. The problems are ``synthetic'', i.e., a TopCoder employee or member creates the problems in order to challenge the competitors and derive their skills ratings. Mike Lydon, chief technology officer for TopCoder and the principal designer of the contest framework, explains:

> ``The problems we pose in the SRM contest are quite demanding and require more than the average amount of mathematical and computer science knowledge along with creativity in exploiting and developing various algorithmic approaches and the ability to translate an abstract problem statement into functional code in 75 minutes or less. To enable consistent testing and ratings of our member base, we design our problems so that we can assess skills in areas as diverse as computational biology, graph theory, image analysis and feature extraction, text mining and semantic analysis, and graphical rendering, amongst many more.

Algorithm contests are held on different days of the week and times of day to encourage participation by TopCoder's global membership. TopCoder advertises contest dates and times well in advance through personalized emails and website announcements. Registration to participate in a contest begins three hours before the start time and closes five minutes before the contest starts. During this period, registered contestants can choose to wait in a virtual chat room, engage in banter with other contestants, and browse information about other registered competitors to get an idea of the skill levels and numbers of those competing in the event. An imprecise measure of skill is readily available in that each coder is listed by handle[7] , along with a color code that provides a rough indicator of the coder's programming ability, based on TopCoder's skills ratings (described in more detail in the next subsection). Participants can obtain more detailed information by looking up another coder's public profile on TopCoder, which prominently displays a coder's skills rating and the percentile in which this rating places the coder relative to others).

Competition occurs in two divisions, I and II, based on participants' skills ratings. Division I consists of more experienced individuals that have higher ratings of 1,200 or above. Division II consists of novices, unrated, and lower-skilled software developers. Developers compete only against others in their division. Our empirical analysis focuses on Division I. This ensures that our analysis of the impact of competition on performance is not influenced by novices learning how the contests work, individuals casually trying out the platform, or low skilled developers who may have difficulty in the competitions regardless of the number and skill distribution of competitors.

All registered competitors who have logged into the TopCoder platform, typically in the hundreds, are placed into virtual competition rooms of around 20 competitors. TopCoder places competitors in rooms with others in the same division; room placement within divisions is random. TopCoder's decision to divide registrants into competition rooms of approximately 20 was driven by early feedback from members, who noted that too large a

[7] A competitor's handle is the unique pseudonym used to identify the competitor on the TopCoder platform. Most online communities use a similar system.

competition room was intimidating and discouraging. TopCoder also discovered that keeping the room size relatively small kept incentives high, because more contestants had a chance to win and place. Importantly, TopCoder distributes the prize purse (if any) based on scoring well within a room. Prior to the start of the contest, a coder does not know which other contest registrants will be in his (or her) competition room. Once the contest begins, a coder has access to a sophisticated ``heads up'' display that provides information about the skill level of each competitor in the room, access to competitors' profiles, a real-time update that shows which competitors have submitted solutions to which problems, and a live scoreboard showing provisional points awarded per problem to each competitor.

The contest format tests a software developer's ability to write code that accurately solves each problem while at the same time rewarding programming speed. Each contest has one easy problem, one problem of medium difficulty, and one hard problem. The point value of each problem indicates its difficulty. The most common distribution of point values is 250, 500, and 1000 for the easy, medium, and hard problems, respectively, although points per problem differ across contests due to differences in problem difficulty, as does the total possible number of points per contest.

Participants have no information about the problems or point values until the contest officially begins. Once the contest starts, participants see the point values for each of the three problems. The participants can then choose to ``open'' any problem in any order and to submit solutions to problems in any order as well. However, as soon as a coder opens a problem by clicking on the problem statement link, the points available to that contestant for the problem start to decline until the individual submits a solution for analysis and testing. Typically, competitors open the three problems in order of difficulty, from easiest to most difficult. If a contestant opens more than one problem simultaneously, the scores for all of the problems start to decline in the same manner until submissions are made.

In order for a submission to be accepted for further evaluation, the submission must compile and have the proper structure to accept input and produce output.[8] Thus, accepted submissions contain no syntax errors or incorrect function names. Once a coder's submission has been accepted, he receives provisional points based on the length of time to submission. If a coder opens a problem but does not submit a solution, the coder receives zero points for that problem.

After the 75 minute coding phase, a 15 minute challenge phase ensues. During this period, a coder may challenge the correctness of accepted submissions. For each challenge, a coder submits a test case (an input or set of inputs to the program) for which an accepted submission by another competitor will produce incorrect output.[9] If a challenge is accurate, the challenger receives 50 points while the competitor loses all points for the submission. If the challenge is inaccurate, the challenger loses 25 points. On average, individual competitors issue challenges in about one-third of contests, typically issuing one or two challenges when they do.

After the challenge phase, the contest moves into an automated testing phase. TopCoder subjects each submission to an automated barrage of test cases and corner conditions to ascertain whether the submission contains logical errors. If a submission fails even one test case, the automated system immediately removes all points provisionally awarded for that submission. Because incorrect submissions receive zero points, in deciding how much time to spend working on a problem before submitting a solution, a coder must balance the reduction in points as time spent working increases against the possibility that more time spent working could increase the likelihood of a correct submission.

To calculate total points per contest for each coder, TopCoder sums the number of points for all correct submissions as well as challenge points earned.

2.2 Rating System

TopCoder assigns a skills rating to each of its members based on points earned in algorithm contests. The rating is an integer value that is updated at the end of each contest in which a member participates.

[8] Checking the syntactical functionality of a software code is similar to the ``hello, world'' test conducted by novice programmers. See http://en.wikipedia.org/wiki/Hello_world_program.

[9] If a challenge is correct, TopCoder incorporates it into the suite of test cases used in the subsequent automated testing phase.

TopCoder uses an Elo type of rating system based on rank order performance (in terms of total points earned per contest), similar to the system used in chess and many sports. In this widely-used type of rating system, the underlying model of contest performance is not strategic; individual performance is presumed to depend only on ability relative to other competitors and random noise. The system is designed to uncover a competitor's true ability when contest performance is a noisy signal of ability. Here we briefly describe the TopCoder rating system. Appendix A provides a fuller explanation of the system.

To calculate a coder's rating at the end of each contest, the rating system first converts the total points scored by each coder into an integer rank in the contest. The system also generates a predicted rank for each competitor, based on a comparison of his pre-contest rating with the pre-contest ratings of other competitors. Both a competitor's actual and predicted rank are adjusted for the number of competitors in the contest, and predicted rank is also adjusted for the variability of a coder's prior ratings; these adjusted ranks are converted to their values in an inverse standard normal distribution. Importantly, a coder's rating depends on where his rank in the contest lies in comparison to his predicted rank based on past contests. For example, if a coder's past history predicts that he will have the 10th highest score in a contest, his rating will decrease if he places 11th. Conversely, his rating will increase if he places 9th. Thus, the system ensures that a less highly rated competitor will not have his rating fall simply because he scores below more highly rated competitors. In fact, a lower-rated competitor needs the presence of higher-rated competitors in order to have a chance of improving his rating.

The rating system is designed so that a coder's pre-contest rating is the primary determinant of the post-contest rating. The system attaches less weight to the difference between actual and predicted rank for coders with more contest experience. In addition, a coder's post-contest rating cannot exceed his pre-contest rating by more than a set value, which is an inverse function of the number of times that a coder has been rated. These features insure that the ratings of coders with more contest experience change less over time than do the ratings of less experienced coders.

2.3 Incentives and Motivations

Our interviews with TopCoder executives and elite members suggest a number of motivations for participating in the algorithm contests. The most obvious relates to the potential to win prize money. In the early days of the platform, TopCoder grew its member base by offering money to the ``Top Coders'' who were able to win contests. This established the TopCoder platform and its rating system as a legitimate avenue for both testing and recording skills in the software community. After TopCoder had established a critical mass of developers, the company gradually withdrew monetary awards for participating in algorithm contests and instead provided prize money largely through client-specific contests. In total, TopCoder has distributed over $1 million in prize money to its algorithm contest participants. In all, about 20 percent of the contests in our data have a monetary reward, with an average of approximately $1,300 in prize money per contest. Prizes are awarded by room, usually to the top two competitors; coders that win a prize receive an average of $51.

Another prominent reason to compete in TopCoder algorithm contests is to receive a TopCoder rating. Because the rating measures programming ability, a number of software development firms request that job applicants get a TopCoder rating before applying. Some firms (e.g., Google, Facebook, Microsoft) sponsor algorithm contests to signal the importance they place on the TopCoder rating. In fact, TopCoder receives more referrals to its website from Google's web page for job applicants than from any other source. One TopCoder member noted that obtaining a high rating is non-trivial -- requiring familiarity with many types of algorithms, quick thinking, strong work ethic, attention to detail, and overall ``smarts''. This member also noted that some of his friends had obtained jobs at high profile technology firms because of their TopCoder rating. Hence, the rating possesses value through improved job prospects and earnings.

In addition to obtaining a TopCoder rating, many developers compete in algorithm contests in order to learn and become better programmers. Developers gain experience during the competition itself, and can learn from other participants in post-competition discussion. TopCoder makes submitted solutions available for competitors to view, and members typically discuss and dissect the solutions after a competition. After each contest, members create a full commentary on each problem and the various solution approaches used in submissions. Members

across the entire skills distribution note the advantage of learning through competitions. One highly-rated member, who prominently has displayed his TopCoder rating on his CV, notes on his resume page:

> ``I regularly participate in and organize a number of programming comptetitions [sic]. As a result, I am familiar with a vast number of algorithmic problems from all areas of computer science. Among the thousands of people who participate in these competitions, I am consistently ranked in the top 50 worldwide. Frankly, I believe that these competitions have taught me more about computer science and programming than all of the university courses. (Ivor Naverinouk, engineer at Google)

Finally, TopCoder members and executives report that the competition format is itself motivating. One member stated (Lakhani, Garvin, and Lonstein, 2010): ``To be successful at TopCoder, you must ask yourself, `Are you a competitor?' You need to be able to thrive on competition; you can't be scared of it.'' TopCoder's founder and chairman, Jack Hughes (2012) remarked that: ``Competition - in games, sports, intellectual exercises (chess, science and math) - is motivating because it is at the core of how we improve. It is difficult to improve at something unless you measure it. Once you measure, you are competing - even if only with yourself - against the clock or to get a better grade for instance. There is real value in simply trying to do something.'' Much as many people join informal, pick-up games of football or basketball, developers enjoy the competition in TopCoder events, especially algorithm contests, as a way to exercise and demonstrate their software skills.

The strength of these various incentives and motivations varies by contest and competitor. Most algorithm contests, for example, do not offer monetary prizes. In addition, as noted above, pre-contest ratings are the primary determinant of post-contest ratings for experienced participants, who make up the majority of competitors in the algorithm contests. Certainly, sustained increases or decreases in final points over multiple contests will change an individual's rating. But for most competitors, the rating system insures that points earned in a single contest do not have a large impact their ratings. As a result, ratings provide a strong incentive mainly for new members and for those with little prior experience. Thus, for many coders, monetary rewards and ratings do not explain why individuals choose to compete in these contests. Rather, opportunities to learn, as well as a desire to exercise programming skills and demonstrate them to others, appear to be strong motivating factors.

3 Dataset, Variables, and Estimation Strategy

Our data analysis focuses on the years 2004-2007. In the years prior to this, TopCoder executives experimented with contest formats and room assignment algorithms before settling on a stable approach to the algorithm contests. This dataset is an unbalanced panel, as not all TopCoder members compete in every contest. As noted earlier, we focus on Division I, which comprises the more highly skilled software developers.

Our data contain records on over 4000 participants active in 181 events. To assess the impact on performance outcomes of the number and skill distribution of competitors, we analyze competition at the room level. During a competition, the information that coders receive about their competitors occurs at the room level. As noted earlier, TopCoder provides a vivid representation of the number and abilities of competitors in a room, and an up-to-date scoreboard that indicates submissions by each competitor in the room, as well as provisional points awarded for each submission. In addition, the distribution of prizes (if any) and post-coding challenges occur within rooms.

3.1 Outcome Variables

As noted earlier, TopCoder measures the problem-solving effectiveness of a participant by summing the total points earned at the end of a contest, denoted by the variable final points. This is the dependent variable in our initial analysis of the effect of increased competition on performance outcomes. In a subsidiary analysis, we decompose final points into those for the first (easy), second (medium difficulty), and third (hard) problems, and the challenge phase. Final points depend heavily on the amount of time spent working on problems, the correctness of submissions, and points earned during the challenge phase. We use these as dependent variables in an analysis of the mechanisms that underlie the performance outcomes that we observe. The three variables are: minutes spent working, an observable measure of effort per problem: challenge issued, a dummy variable indicating whether a

coder issued a challenge, another observable measure of effort; and incorrect submission, a dummy variable that indicates whether a submission failed a challenge or system test due to a mistake in coding caused by a logical error.

3.2 Explanatory Variables

3.2.1 Number of Competitors in the Room

The number of competitors in a virtual contest room, number in room, is our first explanatory variable. Figure 1 illustrates the variation in the number of competitors in our data. The mean number of competitors per room is 18.6, with 85 percent of rooms having 17-20 competitors, providing a 15-20 percent variation in the number of competitors.

[Figure 1 about here.]

Two factors lead to variation in the number of competitors in a contest room: 1) TopCoder's room allocation system, and 2) coders that register for a contest but do not compete. As coders register for a contest, TopCoder aims to fill each virtual competition room with 20 contestants. Participants do not arrive in even groups of 20, however, causing a mathematical indivisibility problem for the room allocation system. The system deals with this problem by trying to create rooms that have roughly the same number of competitors, with no large imbalances between rooms. In addition, during the three-hour registration period, some members who have signed up may decide not to compete. If some registered contestants do not show up at the start of the contest, it affects the number of competitors in a room, because it is too late for the room allocation system to adjust for their absence.

As noted earlier, coders do not receive their room assignments until the contest begins. Although the room allocation system assigns ``no-show'' members to a room during the registration period, other competitors have no knowledge of their presence. No-show members also do not know the identity of the competitors in their room or the nature of the problems before a contest begins. While we cannot directly observe the reasons why members do not show up after registering, investigation of the data indicates that average room sizes are lower on weekdays than on weekends (holding total contest participation constant), suggesting that coders find it harder to predict their schedules during weekdays than on weekends. Our interviews with participants revealed that reasons for not showing up included getting caught up in work activities, social pressures such as friends and spouses needing their attention, and miscalculating when a TV show would be on (e.g., Battlestar Galactica). Hence, the factors leading to variation in the number of competitors in a room appear to be exogenous to the contests themselves. Nevertheless, in the empirical analysis, we control for the total number of competitors per contest, day of the week, and month (e.g., coders may have greater amounts of vacation time during certain months).

3.2.2 Identifying Dominant Competitors (``Superstars'')

Highly dominant competitors make it difficult for others to rank at the top in a contest. Brown (2011), for example, identifies a single individual -- Tiger Woods (when he was highly dominant) -- as a superstar, and uses his presence or absence in a tournament to identify a superstar effect. In our context, we identify a set of overwhelmingly dominant competitors whose abilities indicate that they are likely to score well above other competitors in the room. The number of these ``superstars'' in a room is our second explanatory variable.

The procedure used to identify superstars begins by generating a predicted score (final points) for each competitor in each contest. To do this, for each year in our sample (2004-2007), we first estimate the following model, using data from all contests in the previous three years:

$$p_{it} = \alpha + \beta \text{Rating}_{it} + \nu \text{Rating}_{it}^2 + \gamma X_t^{contest} + \epsilon_{it} \quad (1)$$

- p_{it}: final points for coder i in contest t
- Rating_{it}: pre-contest rating

- Rating_{it}^2: pre-contest rating squared
- $X_t^{contest}$: contest-specific controls

Coder ability, measured by a coder's pre-contest TopCoder rating, is likely to be a key determinant of final points. In addition, we include several control variables. The squared ratings help control for non-linearity in pre-contest ratings.[10] Contest-specific controls, which improve the precision of the model, include the number of competitors in the contest, the point value of each problem, an indicator of whether prize money was available, year of the contest, and month and day of the week dummy variables.

For each year in the sample, to generate predicted scores for each competitor in a given contest, $\mathrm{E}\, p_{it}$, the estimated coefficients from equation 1 are applied to data for each coder in that contest. The standard deviations of these fitted values are stored for each observation. These vary for each observation based on the particular values of the covariates associated with that particular coder and contest. We then generate a predicted performance interval for the final score of each competitor in the contest, as follows:

$$[\mathrm{E}\, p_{it} - 4 \times s.d._{it}, \mathrm{E}\, p_{it} + 4 \times s.d._{it}]$$

The prediction interval for the final score of each competitor in the contest ranges from four standard deviations below the point estimate to four standard deviations above the point estimate, encompassing a relatively large range of potential performance per competitor. These prediction intervals are intended to capture coders' ``competitive proximity'' to other contest participants. Two competitors whose intervals overlap could reasonably expect to place near one another in a contest and are considered close, as illustrated in Figure 2.

[Figure 2 about here.]

Competitors A and B may also be indirectly proximate if A's interval overlaps that of another competitor whose interval overlaps that of B. In order to account for this situation (and further indirection), we take the union of all prediction intervals, as shown in Figure 2. All competitors in a union of predicted intervals are considered part of the same competitive group.

Figure 3 illustrates the process of assigning competitors to competitive groups for an actual room in our dataset. Group numbers are assigned from the lowest expected scores to the highest, starting at 1. If a room contains at least two groups, superstar competitors are those in the highest competitive group. Any individual not in the superstar group is considered a non-superstar.

[Figure 3 about here.]

This procedure for identifying superstar competitors labels 541 competitors as superstars in at least one contest in our sample. The average room has 1.31 superstar competitors, with a median of 1. Ninety-three percent of rooms have 3 or fewer superstar competitors; 13 percent of rooms have only one competitive group and thus have no superstars. On average, a competition room has 2.93 groups, with a median of 3 groups per room. Figure 4 and Figure 5 display histograms of the number of superstars per room and the number of groups per room, respectively.

[Figure 4 about here.]

[Figure 5 about here.]

In order to simplify presentation of the results, our analysis includes only observations for non-superstar competitors. All results hold when using the full dataset; there are no changes in levels of significance and point estimates are very close in magnitude to those presented here.

[10] This turns out to have little impact on predicted scores. We omit it in our primary specification in the main part of our analysis of the effects of increased competition.

3.3 Estimation Approach

We seek to estimate the impact on individual performance of increased competition in the contest overall and from competitors of different types, especially highly dominant ones. We estimate a linear model in which the performance outcome for each competitor in each contest depends linearly on numbers of competitors in total and of different types, as well as on other separable factors such as coder ability and specific features of the contest environment. We also include individual competitor (coder) fixed effects; the model thus exploits performance variation within individuals across contests. In principle, random assignment of competitors to rooms implies that unobserved coder characteristics do not vary systematically with our explanatory variables of interest. Nevertheless, we include individual competitor fixed effects in order to preclude bias due to unobserved heterogeneity of coders.

Our primary specification is a linear model with competitor fixed effects that regresses final points for each competitor in each contest, p_{it}, on measures of the number of competitors in the room (total, superstar, and/or non superstar), as described in the next section; the specification also controls for coder ability and specific features of each contest. For purposes of illustration, in Equation 2 below, N_{it} denotes the total number of competitors in a room faced by competitor i in contest t. To control for ability, we use a coder's pre-contest TopCoder rating. We also control for features of the contest environment: the point value of each of the three problems in a contest, the number of competitors in the entire contest, a dummy variable indicating whether prize money was offered in the contest, the year of the contest (2004-2007), and dummy variables for day of the week (Saturday omitted) and month (June omitted). The variable μ_i indicates individual competitor fixed effects.

$$p_{it} = \alpha + \beta N_{it} + \nu X_{it}^{ability} + \gamma X_{t}^{contest} + \mu_i + \epsilon_{it} \tag{2}$$

Because competitors self-select into the TopCoder member base and into each contest, this could bias our estimated coefficients. In particular, self-selection might affect the distribution of coder ability in a contest. If a particular contest draws a larger proportion of highly skilled competitors than usual, although this would not cause the total number of competitors per room to increase, it could lead to a larger number of superstar competitors in a room on average (since TopCoder assigns competitors randomly to rooms). Examination of the data reveals no difference in the distribution of coder ability per contest associated with month or day of the week. Indeed, the only factor that appears to be correlated with the distribution of coder ability per contest is prize money: contests that offer prize money tend to have a larger number of total competitors because these contests draw a larger number of high ability coders. The inclusion in Equation 2 of the indicator for prize money helps to control for possible sample selection bias from this source. This variable also helps to control for any shift in competitor behavior due to the potential to obtain a monetary reward.

Table 11 in B provides descriptive statistics for all variables used in our analyses.

4 The Effects of Competition

In what follows, we use the specification in Equation 2 to first estimate the impact of the total number of competitors on individual performance, so as to assess whether an N-effect holds. We then add the total number of superstar competitors to the analysis. In addition, we decompose the total number of competitors into the number of superstars and the number of non-superstars. Then we conduct more fine-grained analysis to assess whether the effects of increased competition vary with the ability of competitors, and to further ascertain which competitors may be driving our overall results. We also conduct robustness tests of this analysis using alternate specifications, before examining mechanisms that may underlie our results.

4.1 The N-Effect

As noted earlier, tournament theory implies that an increase in the number of competitors lowers individual effort. In order to check for the presence of an N-effect, we regress final points on the total number of competitors in

the room (N_{it}), using the fixed-effects specification in Equation 2. Table 1 reports the results. As predicted by theory and consistent with prior empirical studies, we find a significant but small negative effect on performance outcomes as the number of competitors in the room increases. On average, final points earned by a competitor fall by about 2 points for each additional competitor in the room. The magnitude of the effect is similar to that seen in earlier work with this dataset Boudreau, Lacetera, and Lakhani (2011).[11]

[Table 1 about here.]

4.2 The Superstar Effect

In tournaments, theory suggests that the presence of competitors of significantly higher ability (superstars) will reduce the efforts of those with lower ability (Szymanski and Valletti, 2005; Brown, 2011). We therefore extend the specification in Equation 2 by including the number of superstar competitors in a room faced by competitor i in contest t (SS_{it}).[12]

$$p_{it} = \alpha + \beta_1 N_{it} + \beta_2 SS_{it} + \nu X_{it}^{ability} + \gamma X_t^{contest} + \mu_i + \epsilon_{it} \quad (3)$$

This specification is motivated by Brown's (2011) study, which compared performance in golf tournaments with and without Tiger Woods (the superstar). In golf, the total number of competitors per tournament is always the same and the number of superstars switches between zero and one, depending on whether or not Tiger Woods competes. In a similar spirit, we estimate the impact of a larger or smaller number of superstars holding the total number of competitors constant. Thus, the superstar effect in Equation 3 can be interpreted as the impact of a competitor switching from non-dominant to dominant, holding the total number of competitors per room constant. In this specification, consistent with empirical evidence in Brown (2011) and other studies, replacing a non-superstar with a superstar in the room leads to a reduction in final points earned by each competitor of about 2.5 points, as shown in Table 2.

[Table 2 about here.]

Unlike in golf and other sporting events, the total number of competitors in TopCoder tournaments varies. Thus, the total number of competitors per room may increase due to an increase in the number of superstars, an increase in the number of non-superstars, or both. Moreover, the overall effect of increased competition may depend on which types of competitors cause the total number of competitors to vary. A variant of Equation 3 captures these effects by decomposing the total number of competitors into the number of superstars (SS_{it}) and non-superstars (NSS_{it}):

$$p_{it} = \alpha + \beta_1 NSS_{it} + \beta_2 SS_{it} + \nu X_{it}^{ability} + \gamma X_t^{contest} + \mu_i + \epsilon_{it} \quad (4)$$

The specification in Equation 4 provides an estimate of the effect of adding a superstar to the room while holding the number of non-superstars constant, and vice versa, which is a natural interpretation of the effects of increased competition in the TopCoder context. In contrast, Equation 2 constrains the effect of adding one more superstar to be the same as the effect of adding one more non-superstar, which presumes that the impact of adding a competitor to the room is the same regardless of that individual's ability. In Equation 3, the estimated coefficient on the number of superstars also does not provide an accurate estimate of the full effect of adding one more superstar to the room: the estimated coefficient on N reflects the effect of adding one more superstar when it is constrained to equal that for a non-superstar, and the estimated coefficient on SS reflects any additional effect of one more superstar beyond that captured by the estimated coefficient on N. Although Equation 3 and Equation 4 are similar, Equation 4 more easily provides estimates of the full superstar effect, and more importantly, makes

[11] Boudreau, Lacetera, and Lakhani (2011) find a somewhat higher effect of approximately 5 points per competitor, but at the room-problem level rather than the individual-contest level and under a different specification.

[12] In our dataset, the number of superstar competitors per room is not particularly correlated with the total number of competitors per room, with a coefficient of correlation of 0.02.

it easy to tell which types of competitors account for the overall effect of increased competition on performance. Therefore, we base our subsequent analyses on the specification in Equation 4.[13] As shown in Table 3, for this specification, adding a superstar to the room leads to a reduction in final points per competitor of 5.7, and adding a non-superstar leads to a reduction in final points of 2.6.

[Table 3 about here.]

In summary, consistent with previous work, we find evidence of small but significant performance losses as the total number of competitors increases. We also find that increased competition from both superstars and non-superstars has a negative effect on performance, but the superstar effect is substantially greater. This provides new evidence that at least in some settings, the overall effect of increased competition may in fact reflect a substantial superstar effect.

5 Heterogeniety in the Response to Competition

The effects just identified apply to the average participant in a contest. However, prior research suggests that the reaction to competition may vary with the ability of competitors. For example, Brown (2011) found that in PGA golf tournaments, Tiger Woods had an effect on the top half of the competitor field in terms of ability, but not the lower half.

Figure 6 depicts the impact of increased competition for each TopCoder rating in our dataset. To generate this figure, we use a two-stage estimation process. In the first stage, we regress final points against our control variables, using a linear specification with individual competitor fixed-effects (that is, Equation 4 without the number of superstars and non-superstars).[14] In the second stage, we use a locally-weighted OLS specification to regress the first-stage residuals (excluding fixed effects) against our measures of competition, namely, the number of non-superstars and the number of superstars in the room. The second-stage regression is run separately for every rating in the dataset, from 1200 to 3754. The regression uses a triangular kernel to weight the observations at each rating; a bandwidth of 300 rating points was used to generate the plots (see Yatchew (1998) and Greene (2003) for additional discussion of the choice of kernel and bandwidth).[15] The coefficient estimates and standard errors for our competition measures from each regression (on the vertical axis) are then plotted against each rating (on the horizontal axis) in Figure 6.

[Figure 6 about here.]

The plots in Figure 6 indicate large negative effects for competitors in approximately the 83rd to 94th percentiles of the ability distribution. For competitors with lower ratings (below about 2050), the negative effects are much smaller. At the far right of the plots, competitors with very high ratings show some evidence of an increase in score as the number of superstars and non-superstars increases. However, there are few competitors in this range and standard errors grow quite large.

[13] Note that because $N = NSS + SS$, the estimated coefficient on NSS in Equation 4 is essentially the same as the estimated coefficient on N in Equation 3. In Equation 3, $\beta_1 N_{it} + \beta_2 SS_{it} = \beta_1(NSS_{it} + SS_{it}) + \beta_2 SS_i t = \beta_1 NSS_{it} + (\beta_1 + \beta_2) SS_i t$.

[14] In estimating the relationship between ratings and competition effects, we tried three additional approaches: dividing ratings into 10 regions with indicator variables, one-stage kernel estimates using OLS, and the two-stage differenced approach used in Yatchew (1998). All show a similar pattern of behavior. In the two-stage approach that we present, the first stage estimates may be biased if the competition effects are correlated with our control variables (see Yatchew (1998) for a discussion). However, one-stage OLS and the differenced approach do not allow for the inclusion of competitor fixed-effects. As the first-stage coefficient estimates with our control variables were similar to the estimates obtained from the differenced approach, we felt that controlling for competitor heterogeneity in the first stage was the best option.

[15] Given an infinite supply of data, the most accurate approach would be to obtain coefficient estimates for each rating in the dataset using only data for competitors with that rating. Data limitations make this infeasible. Therefore, to obtain the coefficient estimates at each rating, we include data on ``nearby'' competitors whose ratings place them within an estimation window that is 300 rating points wide, termed the ``bandwidth'', centered on the rating for which the estimates are to be generated. A relatively wide bandwidth incorporates more data and tends to result in a smoother estimated relationship, but too large a bandwidth may include observations that are not relevant for the rating in question. We tried several bandwidths from 50 to 300 rating points, and all showed a similar relationship. In addition to the choice of bandwidth, which determines which observations are included in the estimation at each rating in the dataset, the observations within the estimation window can be weighted so that observations closer to the rating in question are given more weight than those toward the edges of the estimation window. Data points closer to the rating in question are more informative about the relationship being estimated. The choice of kernel determines this weighting and provides another tool to smooth the estimated relationship. Our plot uses a triangular kernel so the weight falls linearly with the distance from the center of the estimation window. In practice, the choice of kernel is rarely crucial in obtaining estimates (Greene, 2003).

Closer examination of the data underlying the plots in Figure 6 enables us to define a ``performance deterioration zone'' (PDZ) that encompasses the strongest negative reactions to increased competition. This zone, with ratings in the range of 2051 to 2414, encloses the largest magnitude of negative effects seen in the kernel estimates, as depicted in Figure 6. Although the boundaries of this range are somewhat blurred due to the use of kernel-based methods, the analysis reported next is not sensitive to the precise location of the upper and lower boundaries of the PDZ.

To further investigate the sources of these effects of increased competition, we decompose the final points of each competitor into points earned for each of the three problems and points earned through challenges. Figure 7 depicts plots for points earned on each problem and through challenges, using the same kernel-based technique described earlier. The response seen in total points is mechanically the sum of the responses for these four subscores. As shown in Figure 7, most of the response to increased competition revolves around the third problem. The points earned from problems 1 (easy) and 2 (medium difficulty) and from challenges show little response to competition across the ability distribution.

[Figure 7 about here.]

The information from the kernel analysis enables us to estimate regressions that more precisely assess how competitors of different levels of ability react to increased competition. We use indicators of whether a competitor is below, within, or above the PDZ, and interact each indicator with the number of superstars and the number of non-superstars in the room. Table 4 presents estimates using the new specification with final points as the dependent variable, as well as with points earned for each of the problems (first, second, and third) and the challenge score as dependent variables.

[Table 4 about here.]

As anticipated, competitors in the PDZ show significant negative effects of increased competition on final points, which drop by about 7.5 points per additional non-superstar in the room and by about 20 points for each additional superstar. In addition, competitors whose ratings are above the PDZ show a decrease in final points for each additional non-superstar. These competitors also show an increase in points for each additional superstar, a result to which we return later.

For coders below the PDZ, Table 4 shows that these coders react negatively, albeit less strongly, to increased competition. Like the more skilled competitors in the PDZ, lower ability coders react more negatively to the number of superstars than to the number of non-superstars. This reaction centers on problems 1 and 2, most likely because less skilled competitors concentrate on the less difficult problems. Conversely, more highly skilled coders within and above the PDZ have a significant reaction to increased competition only for problem 3, the most difficult problem. Indeed, we see no significant reactions of the more highly skilled coders for the less difficult problems, which are likely to prove less demanding for these coders and therefore may elicit less of a reaction to increased competition.

In section 7, we examine underlying mechanisms that may drive these performance outcomes. We focus on the third problem, which accounts for a large portion of the effect on performance, and on the most affected group of coders in the PDZ and above the PDZ, as doing so provides the most statistical power for identifying the underlying mechanisms. Narrowing the focus to a single problem reduces the dimensions of competitor actions considerably; essentially, we no longer need to consider all actions in triplicate. However, we emphasize that this selection of focus is driven by the dataset, and may not generalize to other contexts.

6 Robustness

Before proceeding to an examination of mechanisms that may underpin the effects of increased competition, we conduct robustness tests of our primary competitor fixed effects specification. In particular, we examine alternate approaches to controlling for heterogeneity across coders and contests, using the specification just reported in Table 4 for the third problem as the basis for comparison.

Table 5 reports the results of these robustness tests. Column one reports coefficient estimates from an ordinary-least-squares (OLS) specification with no controls. The specification in column two accounts for heterogeneity through the addition of individual competitor fixed effects but without controls for competitor ability or contest features. Notably, in both alternate specifications, the coefficient estimates and significance levels are similar to those in our primary specification, shown in column three for ease of comparison. This suggests that the results reported earlier are highly robust, even without controlling for competitor ability, competitor fixed effects, or specific contest features. In addition, as shown in column five, the inclusion of the number of previous contests that a competitor has entered, which accounts for any potential impact of contest experience on performance, has little impact on the results; the estimated coefficients and significance levels are almost identical to those in the primary specification.

[Table 5 about here.]

As an alternate method of controlling for heterogeneity in contests, we replace the contest controls with contest fixed effects, omitting competitor fixed effects. (Including fixed effects can account for heterogeneity in competitors or contests, but not both simultaneously.) Using contest fixed effects shifts the identification from within competitors, across contests to within contests, across competitors. As shown in column four, the results seen in the primary specification for competitors in the PDZ and above the PDZ continue to hold. The coefficient on non-superstars for competitors below the PDZ becomes significant and has a positive sign, however. This may be an artifact of the room allocation procedure described earlier, which does not adjust for no-shows (coders who register for a contest but do not compete). No-shows are likely to be concentrated among non-superstars, simply because superstars are rare by definition; this in turn will drive variation in the number of non-superstars in the room. Having fewer non-superstars due to no-shows especially benefits lower ability contestants, who are below the PDZ, leading to the positive coefficient.

In addition to robustness tests of our primary specification, we conduct robustness tests involving the superstar effect. First, we replace the variable for the number of superstars with dummy variables indicating whether a room has one, two, three, four, or five or more superstars, and interact each of these dummy variables with the indicators for coders below the PDZ, in the PDZ, and above the PDZ. The results, shown in Table 12 in B, are generally consistent with those reported earlier, although they exhibit some variability due to lower degrees of freedom for each interaction term. Overall, the results suggest that the reaction to intensified competition tends to increase as the number of superstars increases.

One concern with the procedure for identifying superstar competitors is that it may create a mechanical effect on performance. As the number of superstar competitors increases and more of the high ability competitors are labeled superstars, the average ability of non-superstar competitors may fall, leading to a negative effect on performance. In order to check that such a selection effect is not driving the results, we run our analysis on a restricted set of rooms in which the number of dominant competitors is less than k (where k is either 3 or 4), and include in the analysis only competitors predicted to have a rank of more than k-1 (where a higher numerical rank indicates placing less well). For this population, the top k-1 competitors are never included in the analysis, even if they are not superstars; the analysis is run with N-k-1 competitors per room, insuring that the average ability of non-dominant competitors varies only with N and not with the number of superstars. The results, shown in Table 6, are consistent with the results from the full sample.[16]

[Table 6 about here.]

7 Mechanisms and Competitor Actions

As our estimates of the effects of increased competition from both superstars and non-superstars appear robust, we continue the analysis by examining underlying mechanisms that may account for these results. In what follows, we continue to focus on points earned on the third problem.

[16] In order to improve precision of the estimates for the control variables, all competitors are included in the analysis. For each regression on the restricted set of rooms with less than k superstars, only competitors in those rooms are used to estimate the interaction terms shown in Table 6, by interacting these terms with a dummy variable indicating whether an observation comes from a room with less than k superstars.

To begin, Figure 8 displays kernel estimates of the impact of the number of superstars and non-superstars on: 1. points earned on the third problem, 2. the correctness of submissions, and 3. minutes spent working on submissions. The y-axes are scaled so that the vertical distance in correctness and minutes spent working roughly corresponds to the same vertical distance for points on the third problem. The figure suggests that the response to increased competition seen in points on the third problem is strongly related to the correctness of submissions. Whereas time spent working shows little response to increased competition across the ratings distribution, correctness of submissions closely tracks the changes in points.

[Figure 8 about here.]

Regression analysis using our primary fixed effects specification supports the patterns shown in Figure 8. As indicated earlier, we focus on the most affected groups, namely, coders in the PDZ and above the PDZ. Table 7 reports the estimated effect of increased competition on minutes spent working on all submissions for the third problem, and on minutes spent working on correct submissions only.[17] For minutes spent working on all submissions, the only significant reaction comes from coders in the PDZ, who take longer to submit as the number of non-superstars increases. For minutes spent working on correct submissions, again only coders in the PDZ have a significant reaction, but at the 10 percent level of significance and to an increase in the number of superstars rather than non-superstars. Although these estimates suggest some reaction of those in the PDZ, they are inconsistent regarding the cause (superstars or non-superstars), and the reaction to superstars is not highly significant.

[Table 7 about here.]

In contrast to the somewhat inconclusive results for minutes spent working, we find a highly significant increase in incorrect submissions for those in the PDZ in reaction to both superstars and non-superstars, as reported in Table 8.[18] Submissions by this group are incorrect about 2 percent more often for each additional non-superstar and about 5 percent more often for each additional superstar. In addition, competitors above the PDZ, whose scores on the third problem show a positive response to additional superstars, are estimated to be incorrect about 1 percent less often for each additional superstar, although the effect is statistically insignificant. A Wald test, however, confirms that the marginal effect for these competitors is significantly below that of competitors in the PDZ, who are most strongly affected.

[Table 8 about here.]

7.1 Causes of Performance Deterioration

These results suggest that the negative effects of increased competition are associated with a greater likelihood of making an error on the most difficult problem, rather than with an increase in time to submission. This effect is concentrated among coders in the PDZ who are near but not at the top of the ability distribution. Potential causes of increased errors include deliberate (conscious) actions as well as unconscious reactions in the face of increased competition. Next we assess whether deliberate choices by competitors, such as increasing the riskiness of contest strategy or reducing effort, may explain the performance decrements that we observe.

7.2 Risk-taking

During contests, competitors may take strategic risks. For example, consider a losing football team that runs a ``Hail Mary'' play in the last seconds of a game to try to eke out a victory. Although risky, if the play succeeds, the team wins the game. Similarly, in algorithm contests, consider the extreme case of a competitor who cares only about placing first in the room. This is likely to cause the competitor to spend less time working on a problem and to

[17] These analyses include only coders that opened the third problem.

[18] In order to improve precision of the estimates for the control variables, all competitors are included in the analysis. Competitors who did not open the third problem are coded as having an incorrect submission. Only competitors who opened and submitted a solution to the third problem are used to estimate the interaction terms shown in Table 8, by interacting these terms with a dummy variable indicating whether there was a submission for the third problem.

submit a solution earlier than otherwise, which increases the provisional score (before correctness is evaluated). Although this strategy increases the risk that the submission is incorrect and receives zero points, it will increase the points awarded for the problem if it is correct--enabling the competitor to place more highly in the room. More generally, if competitors employ a riskier strategy of this sort, they will submit solutions more quickly with some positive probability.

We look for evidence of such an approach in the amount of time spent working on submissions for the third problem. The average time spent working should fall if competitors shift to submitting fast but more error-prone solutions. The second column of Table 7 shows that the only significant effect on minutes spent working on submissions comes from coders in the PDZ in the face of an increased number of non-superstars. This effect is positive rather than negative, however, providing no evidence that competitors switch to riskier strategies in the face of increased competition.

7.3 Observed Effort

Perhaps the simplest explanation for the decrease in performance that we observe is reduced effort by competitors. As the likely placement of a competitor falls due to increased competition, the expected benefit of exerting effort falls as well. A competitor then may find it in his interest to reduce opportunity costs by reducing effort.

For evidence of reduced effort, we examine three indicators that competitors stop work early, essentially dropping out of the contest. First, we examine whether competitors open the third problem. If a competitor fails to open the third problem, he may no longer be trying to earn points in the contest; he may also still be trying to solve one of the other problems. The former would indicate a ``drop out'' effect, while the latter would not. As shown in Table 7, we do not observe such an effect; an increase in the number of superstars or non-superstars does not significantly affect whether competitors open the third problem.

The amount of time spent working on the third problem (for coders who have opened this problem) provides a second indicator of effort. Significantly less time spent on the third problem may indicate that a competitor has dropped out; it could also indicate greater risk taking, as noted earlier. However, as shown in the second and third columns of Table 7, time spent working on submissions either does not change significantly or increases as the number of superstars and non-superstars increases.

Lastly, recall that each contest ends with a 15 minute challenge phase, in which coders can examine whether the submissions of other coders contain logical flaws. If coders drop out of the contest, we would not expect them to participate in the challenge phase. Hence, the likelihood that a competitor issues a challenge should fall among the group most likely to drop out, namely, those who open the third problem but do not submit a solution. If coders in this group drop out, we would expect to see a negative effect of increased competition on the likelihood of issuing a challenge. As shown in Table 9, for coders in this group, the estimates show essentially no change in the likelihood of issuing a challenge in the face of a larger number of non-superstars. The same holds for coders in the PDZ in the face of a larger number of superstars.[19] The only change in behavior comes from coders above the PDZ, who increase rather than decrease challenges when faced with additional superstars, consistent with earlier results showing that the final points for these coders increase in the presence of a greater number of superstars.

[Table 9 about here.]

In summary, these results provide no evidence that coders reduce their effort as the number of non-superstars or superstars increases. Coders are no less likely to open the third problem, do not spend less time working on it, and are not less likely to issue a challenge in response to increased competition.

7.4 Cognitive Changes

Although we find no evidence that coders reduce observable effort in response to increased competition, points earned in a contest also depend on the correctness of submissions independent of effort. As noted earlier, errors

[19] The interaction terms shown in Table 9 are also interacted with a dummy variable for having opened but not submitted a solution to the third problem.

in accepted submissions indicate logical flaws in submitted code, which are likely to derive from cognitive factors. In order to investigate the possibility that cognitive factors explain the performance outcomes that we observe, we examine whether increased competition affects the likelihood of an incorrect submission for the third problem (conditional on submitting). We control for time spent working on submissions, because greater time spent working may reduce errors.

As shown in Table 10, the results indicate that near-to-the-top competitors in the PDZ are significantly more likely to produce faulty code as the number of non-superstars and superstars increases.[20] For the same amount of time spent working, these coders make about 1.5 percent more incorrect submissions for each additional non-superstar in the room. These coders also make about 4 percent more incorrect submissions for each additional superstar in the room. Hence, even when controlling for observable effort in terms of the amount of time that coders work on solutions, coders in the PDZ make more logical errors as competition increases.[21] Here again, we find evidence that coders react far more strongly to increased competition from superstars than from non-superstars. In addition, we find no evidence that coders above the PDZ make more logical errors in the face of increased competition from non-superstars or superstars.

[Table 10 about here.]

Taken together, the evidence suggests that near-to-the-top competitors in the PDZ do not reduce their observable effort but make more logical mistakes. As noted earlier, a variety of cognitive factors could explain these results. For example, these mistakes could stem from a reduction in cognitive effort. That is, the brain may get tired even as coders continue to work on problems. In an experimental study, Bracha and Fershtman (2012) find that some people may work harder but not smarter under tournament conditions.

As an alternative or possibly complementary explanation, psychological pressure from increased competition may lead to choking that in turns leads to more mistakes. Choking occurs even when incentives for superior performance are high (Baumeister and Showers, 1986). For example, individuals have been found to choke when faced with high financial incentives and competitive stakes (Ariely et al., 2009; Apesteguia and Palacios-Huerta, 2010). In a review of a large amount of evidence in psychology, DeCaro et al. (2011) note that two separate mechanisms cause choking. First, self-consciousness about performing correctly leads to increased attention on the precise steps in learning and executing skills. This in turn disrupts ``procedural'' skill execution that takes place without conscious awareness, commonly seen in sports activities like golf putting, hockey dribbling, and baseball batting. Secondly, distraction caused by undue focus on performing well reduces the amount of working memory available, as in mathematical puzzle-solving (Beilock and Carr, 2001, 2005). In the contests examined here, pressure to perform well could disrupt coders' routine (procedural) approaches to algorithmic problem solving, and a reduction in working memory could affect the ability to solve attention-demanding algorithmic problems.

In contrast to near-to-the-top competitors in the PDZ, coders above the PDZ do not make more logical errors, and they exert greater effort by making more challenges when faced with increased numbers of superstars. These competitors, who have predicted ranks just below superstars, do not appear to suffer from cognitive deterioration in problem solving. As noted earlier, for these competitors, self-confidence stemming from high ability may counterbalance any negative impact of pressure to perform well. In addition, the increase in observable effort in the face of additional superstars is consistent with the argument that those on the edge of winning positions have an incentive to exert maximum effort (Casas-Arce and Martínez-Jerez, 2009).

8 Summary and Conclusions

This study began with the observation that prior research has found that individuals in tournament-style contests perform less well in the face of increased competition, but that studies often lack evidence about the mechanisms that underlie this result, especially in field settings. We provide evidence regarding three mechanisms that may

[20] The interaction terms shown in Table 10 are also interacted with a dummy variable for having submitted a solution to the third problem.

[21] Note that the reported coefficients on the time spent working in Table 10 may be biased. It is possible that time spent working and incorrect submissions are both influenced by an unobserved cognitive variable. However, the coefficients on the competition measures will still be estimated consistently.

account for such a performance decline: reduction in effort, increased risk taking, and deterioration in cognitive processing.

In the algorithmic programming contests studied here, we find that the largest negative reactions to increased competition come from a group of competitors who are near-to-the-top in terms of ability. In contrast to the predictions of tournament theory, we find no evidence that competitors in this group reduce their effort in reaction to increased competition. For example, time spent working on problems, a particularly relevant measure of effort in these contests, does not decrease as the number of competitors increases. We also find no evidence of increased risk taking. Instead, the evidence shows that competitors in this group make more logical errors when faced with increased competition, especially from superstars, suggesting that cognitive factors at least partly account for the decline in performance. We also find that a small group of very high ability competitors (excluding superstars) reacts positively to increased competition from superstars. These competitors exert somewhat greater effort during contests, consistent with economic logic that competitors on the edge of winning may exert maximum effort, and cognitive errors do not increase, consistent with psychological research suggesting that very high ability competitors may not succumb to performance pressure.

In addition to providing evidence on the mechanisms that underpin changes in performance in reaction to increased competition, this study extends prior research in a number of ways. First, the structure of TopCoder contests enables us to go beyond prior empirical research on the N-effect, by distinguishing between increased competition from superstars and non-superstars. We find that an additional superstar has a much more negative effect on the performance of the average competitor than an additional non-superstar. This finding also contributes to empirical research on the superstar effect, which has focused heavily on sporting events, by providing evidence from non-sports contests that increased competition from superstars negatively affects the average performance of non-superstars.

Our study also shows that the ability of competitors affects their reactions to increased competition. Prior research in both economics and psychology suggests that more highly skilled competitors may have the strongest reactions. We find that this holds in online algorithm contests. Although lower ability competitors have a significant (and negative) reaction to increased competition, most of the reaction comes from high ability competitors (excluding superstars). Our results show that these high ability competitors react negatively to an increase in non-superstars. This is not surprising, given that non-superstars include some competitors of high ability, who therefore may cause the rank of a high ability competitor to decline. In addition, like lower ability competitors, these high ability coders react more strongly to increased competition from superstars than from non-superstars. However, as noted above, the high ability coders differ in their reactions to superstars. Most react negatively, but a small group of competitors, who are just below superstars in their abilities, have a positive reaction. Although Brown (2011) does not find a positive reaction of highly skilled players to the superstar Tiger Woods in PGA tournaments, Connolly and Rendleman Jr. (2009) find that high ability players paired with Woods perform better when both are in contention to win a tournament than when they are not. These mixed findings regarding such next-to-the-top competitors suggest that future research is warranted.

Although very high ability coders react positively to increased competition from superstars in the algorithm contests, most competitors react negatively to an increase in both superstars and non-superstars. Strikingly, coders who account for the largest portion of this negative reaction do not reduce observable ``labor'' effort. Instead, we find that these coders make more logical errors when faced with increased competition. Bracha and Fershtman (2012) provide experimental evidence in tournaments that suggests that reduced cognitive effort may play a role. In addition, although psychology research has not examined choking in tournaments, new experimental evidence suggests that choking may be especially relevant in this setting. This evidence comes from DeCaro et al. (2011), who show that the makeup of the pressure situation affects which of the choking mechanisms (disruption of procedural skill execution or a reduction in working memory caused by distraction) comes into play. In particular, pressure from being watched, termed ``monitoring pressure'', leads to disruption of procedural learning and skill execution.[22] In contrast, pressure to earn a reward if a certain outcome is achieved, termed ``outcome pressure'', leads to distraction from the task at hand. Many tournaments contain both types of pressure. Partici-

[22] DeCaro et al. (2011) mention the presence of a mirror or a video camera as creating monitoring pressure from being watched, in addition to watching by other individuals. Thus, being watched in any way results in monitoring pressure.

pants watch and are watched by other participants, which could disrupt participants' procedural skill execution. In addition, almost by definition, tournaments contain rewards for performing well (even if simply ranking highly), which could distract participants from devoting full attention to the task at hand.

More generally, a better understanding of behavioral responses in contests can aid both public policy and contest designers. The use of contests to elicit creative effort and technological innovation has gained renewed interest in both the public and private sectors (Tapscott and Williams, 2006; National Research Council, 2007; McKinsey & Company, 2009; Zients, 2010). With further work to understand what triggers performance losses, contest designers may be able to avoid generating nuisance effects in the contest environment such as choking. While developing a resilience to choking may be beneficial for athletes whose jobs entail participating in competitive sporting events, such resilience is unlikely to be critical for software development, scientific research, and creative skills now sought in online contests. This suggests that sponsors of such contests face the challenge of finding ways to reduce the negative effects of cognitive factors so that contests can better measure ability and provide incentives for performance.

9 References

References

Apesteguia, J., Palacios-Huerta, I., 2010. Psychological pressure in competitive environments: Evidence from a randomized natural experiment. American Economic Review 100, 2548--2564.

Ariely, D., Gneezy, U., Loewenstein, G., Mazar, N., 2009. Large stakes and big mistakes. Review of Economic Studies 76, 451--469.

Arnsten, A. F. T., 2012. Stress signalling pathways that impair prefrontal cortex structure and function. Nature Reviews Neuroscience 13, 410--422.

Autor, D. H., 2001. Wiring the labor market. The Journal of Economic Perspectives 15, 25--40.

Baumeister, R. F., Showers, C. J., 1986. A review of paradoxical performance effects:choking under pressure in sports and mental tests. European Journal of Social Psychology 16, 361--383.

Beilock, S. L., Carr, T. H., 2001. On the fragility of skilled performance: What governs choking under pressure? Journal of Experimental Psychology: General 130, 701--725.

Beilock, S. L., Carr, T. H., 2005. When high-powered people fail : Working memory and ''choking under pressure'' in math. Psychological Science 16, 101--105.

Boudreau, K. J., Lacetera, N., Lakhani, K. R., 2011. Incentives and problem uncertainty in innovation contests: An empirical analysis. Management Science 57 (5), 843--863.

Bracha, A., Fershtman, C., 2012. Competitive incentives:working harder or working smarter? Management Science Forthcoming (5-12), --.

Brown, J., 2011. Quitters never win: The (adverse) incentive effect of competing with superstars. Journal of Political Economy 119 (5), 982--1013.

Brunt, L., Lerner, J., Nicholas, T., 2011. Inducement prizes and innovation. Discussion paper SAM 25 2011, Norwegian School of Economics.

Casas-Arce, P., Martínez-Jerez, F. A., 2009. Relative performance compensation, contests, and dynamic incentives. Management Science 55, 1306--1320.

Che, Y.-K., Gale, I., 1983. Optimal design of research contests. American Economic Review 93, 646--671.

Connolly, R. A., Rendleman Jr., R. J., February 2009. Dominance, intimidation and `choking' on the pga tour. Journal of Quantitative Analysis in Sports 5, 1--32.

DeCaro, M. S., Thomas, R. D., Albert, N. B., Beilock, S. L., 2011. Choking under pressure: Multiple routes to skill failure. Journal of Experimental Psychology: General 140, 390–406.

Fullerton, R. L., Linster, B. G., McKee, M., Slate, S., 2002. Using auctions to reward tournament winners: Theory and experimental investigations. The RAND Journal of Economics 33, 62--84.

Fullerton, R. L., McAfee, R. P., June 1999. Auctioning entry into tournaments. Journal of Political Economy 107 (3), 573--605.

Garcia, S. M., Tor, A., 2009. The n-effect more competitors, less competition. Psychological Science 20 (7), 871--877.

Greene, W. H., 2003. Econometric Analysis. Prentice-Hall, Upper Saddle River, NJ.

Holmstrom, B., Milgrom, P., 1991. Multitask principal-agent analyses: Incentive contracts, asset ownership, and job design. Journal of Law, Economics, & Organization 7, 24--52.

Kahneman, D., 1973. Attention and effort. Prentice-Hall, Englewood Cliffs, NJ.

Knoeber, C. R., Thurman, W. N., 1994. Testing the theory of tournaments: An empirical analysis of broiler production. Journal of Labor Economics 12, 155--179.

Konrad, K. A., 2009. Strategy and Dynamics in Contests. Oxford University Press.

Kremer, M., 1998. Patent buy-outs: A mechanism for encouraging innovation. Quarterly Journal of Economics 4 (6304), 1137--1167.

Kremer, M., Williams, H., 2010. Incentivizing innovation: Adding to the tool kit. In: Innovation Policy and the Economy. University of Chicago Press, pp. 1--17.

Lakhani, K. R., Garvin, D. A., Lonstein, E., 2010. Topcoder (a): Developing software through crowdsourcing. Case Study 9-610-032, Harvard Business School.

Lallemand, T., Plasman, R., Rycx, F., 2008. Women and competition in elimination tournaments evidence from professional tennis data. Journal Of Sports Economics 9, 3--19.

Lazear, E. P., Rosen, S., 1981. Rank-order tournaments as optimum labor contracts. Journal of Political Economy 89, 841--864.

McKinsey & Company, October 2009. And the winner is: Capturing the power of philanthropic prizes. Online. URL http://www.mckinsey.com/clientservice/Social_Sector/our_practices/Philanthropy/Knowledge_highlights/And_the_winner_is.aspx.

Moldovanu, B., Sela, A., 2001. The optimal allocation of prizes in contests. American Economic Review 91, 542--558.

National Research Council, 2007. Innovation Inducement Prizes at the National Science Foundation. The National Academies Press, Washington, DC.

Otten, M., 2009. Choking vs. clutch performance: A study of sport performance under pressure. Journal of Sport and Exercise Psychology 31, 583--601.

Prendergast, C., 1999. The provision of incentives in firms. Journal of Economic Literature 37, 7--63.

Riley, D., 2012. New tiger, old stripes. Gentlemen's Quarterly.

Scotchmer, S., 2004. Innovation and incentives. MIT Press.

Sunde, U., December 2003. Potential, prizes and performance: Testing tournament theory with professional tennis data. Discussion Paper 947, Institute for the Study of Labor, IZA P.O. Box 7240 D-53072 Bonn Germany.

Szymanski, S., Valletti, T. M., 2005. Incentive effects of second prizes. European Journal of Political Economy 21, 467--481.

Tanaka, R., Ishino, K., 2012. Testing the incentive effects in tournaments with a superstar. Journal of The Japanese and International Economies in press., xxx--xxx.

Tapscott, D., Williams, A. D., 2006. Wikinomics: How Mass Collaboration Changes Everything. Penguin, New York.

Taylor, C. R., 1995. Digging for golden carrots: An analysis of research tournaments. American Economic Review 85, 872--890.

Terwiesch, C., Xu, Y., 2008. Innovation contests, open innovation, and multiagent problem solving. Management Science 54, 1529--1543.

Wright, B. D., 1983. The economics of invention incentives: Patents, prizes, and research contracts. The American Economic Review 73, 691--707.

Yatchew, A., 1998. Nonparametric regression techniques in economics. Journal of Economic Literature 36, 669--721.

Zients, J. D., March 2010. Guidance on the use of challenges and prizes to promote open government. Memorandum for the Heads of Executive Departments and Agencies.
URL http://www.whitehouse.gov/omb/assets/memoranda_2010/m10-11.pdf

Appendices

A The TopCoder Rating System

Here we describe the main elements of the TopCoder rating system. The rating system is described in greater detail at: HTTP://WWW.TOPCODER.COM/WIKI/DISPLAY/TC/ALGORITHM+COMPETITION+RATING+SYSTEM. The formula for a coder's rating is:

$$\text{NewRating} = \frac{\text{OldRating} + \text{Weight} \cdot \text{PerfAs}}{1 + \text{Weight}}$$

A coder's rating is updated at the end of each contest to produce NewRating.[23] OldRating is the coder's pre-contest rating. If a coder has never competed in a TopCoder algorithm contest, TopCoder assigns a value of 1200 to OldRating.

Rearranging terms, based on the formula for PerfAs below, yields:

$$\text{NewRating} = \text{OldRating} + \left(\frac{\text{Weight}}{1 + \text{Weight}}\right) \cdot \text{CF} \cdot (\text{APerf} - \text{EPerf})$$

PerfAs is the provisional rating assigned to each coder at the end of a contest.

$$\text{PerfAs} = \text{OldRating} + \text{CF} \cdot (\text{APerf} - \text{EPerf})$$

[23] On the TopCoder website, in the explanation of the rating system, the variable Rating is sometimes used in place of what we term OldRating. We use NewRating and OldRating for clarity.

APerf is the coder's rank order performance in the contest, calculated as a value in an inverse standard normal distribution that adjusts for the number of coders per contest:

$$\text{APerf} = -\Phi\left[\frac{\text{ARank} - 0.5}{\text{NumCoders}}\right]$$

where ARank is the coder's rank in a contest, based on total points per coder and NumCoders is the number of coders in the contest.

EPerf is the predicted value of APerf, based on the coder's pre-contest rating relative to the pre-contest ratings of other contestants:

$$\text{EPerf} = -\Phi\left[\frac{\text{ERank} - 0.5}{\text{NumCoders}}\right]$$

where $\text{ERank} = 0.5 + \sum_i WP_i$. WP_i, or Win Probability, is the probability that the coder will have a higher score than another coder i in the contest. Each Win Probability is calculated based on the pre-contest ratings of coders that entered the contest, adjusted for a measure of the spread of each coder's prior contest ratings, termed Volatility. Coders that have never competed before receive an initial value of 300 for Volatility.

In the formula for PerfAs, CF denotes a ``Competition Factor'' for each contest. CF captures the spread of the pre-contest ratings of coders in the contest, based on both pre-contest Volatilities of the contestants and a measure of the difference between the average pre-contest rating of contestants and individual coder pre-contest ratings. A greater spread of pre-contest ratings results in a higher competition factor, leading to a higher weight on the difference between a coder's actual and anticipated performance. Intuitively, changes in rank order performance in a contest where coders have similar abilities, as measured by pre-contest ratings, are more likely to reflect random factors rather than skill, and therefore receive lower weight in calculating the new rating. Finally, in the formula for NewRating, Weight for each coder is an inverse function of the number of times that the coder has been rated previously. More experienced coders have less weight attached to the difference between their current rank order performance, APerf, and their predicted rank order performance as reflected in EPerf. In addition, a coder's NewRating cannot exceed his or her OldRating by more than a set value termed Cap, which is an inverse function of the number of times that a coder has been rated. The values of Weight and Cap insure that the ratings of more experienced coders change less over time than do the ratings of less experienced coders.

B Large Tables

[Table 11 about here.]

[Table 12 about here.]

Figures

Figure 1: Number of Competitors in Room.

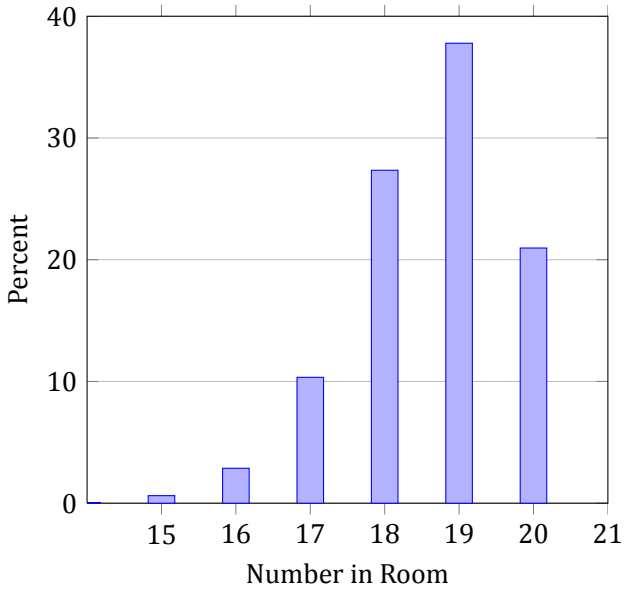


Figure 2: Illustration of how competitive proximity is defined.

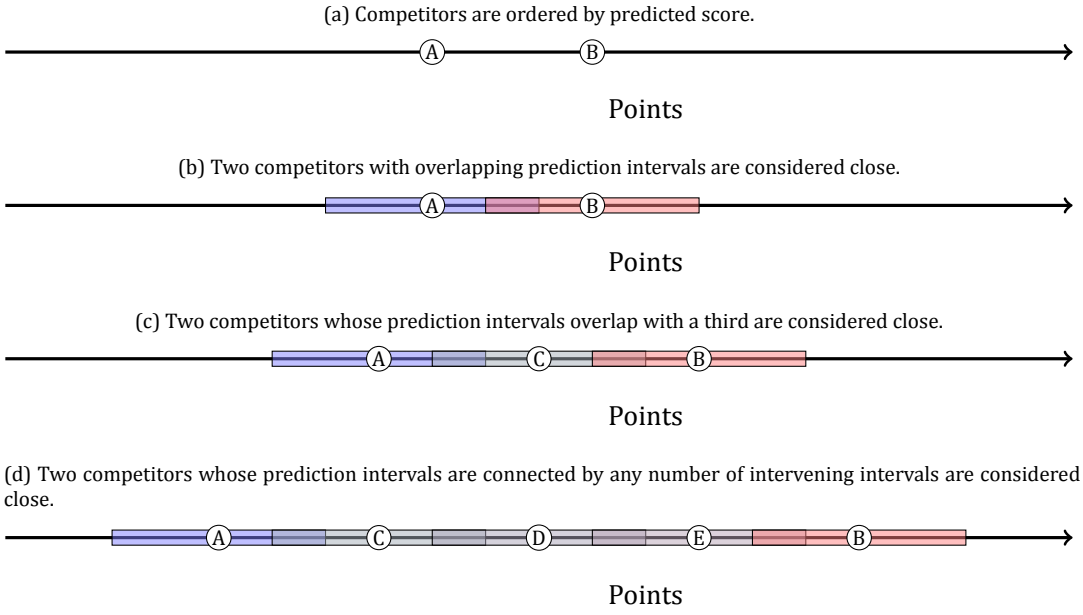


Figure 3: Illustration of the identification of superstar competitors.

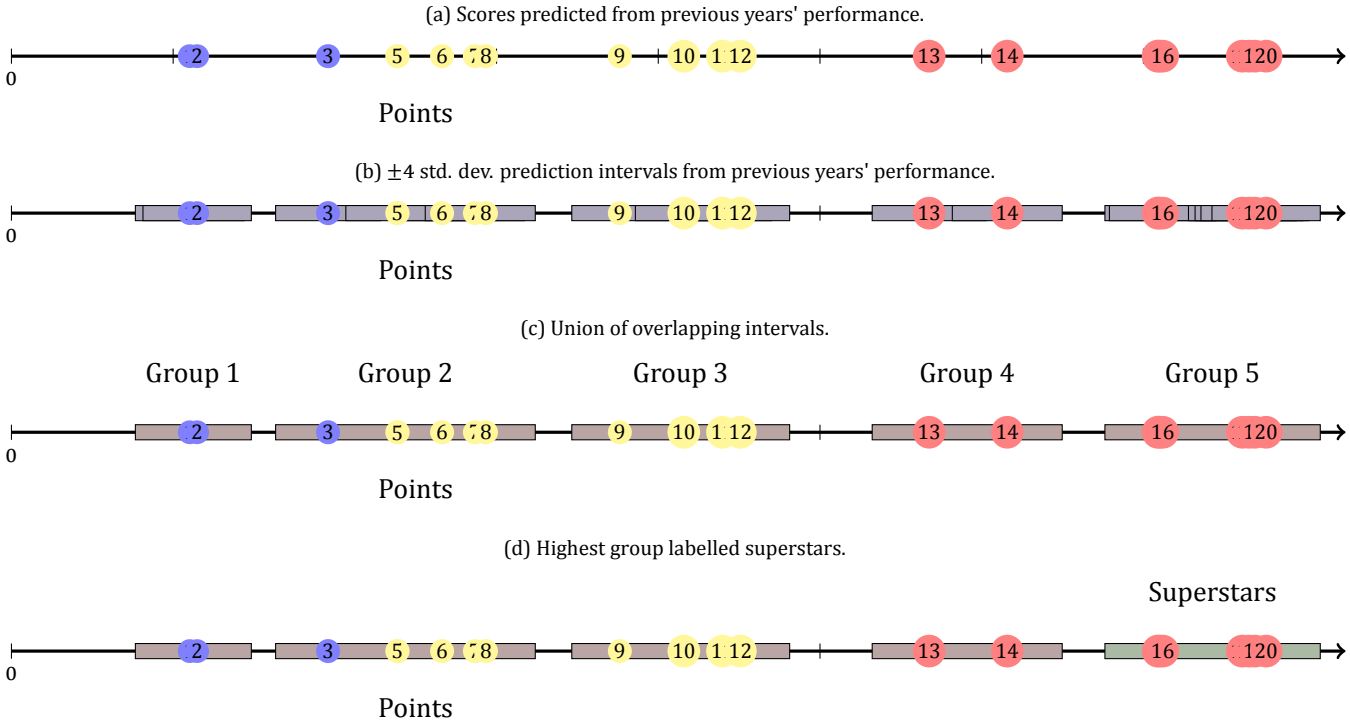


Figure 4: Number of Superstars in competition rooms.

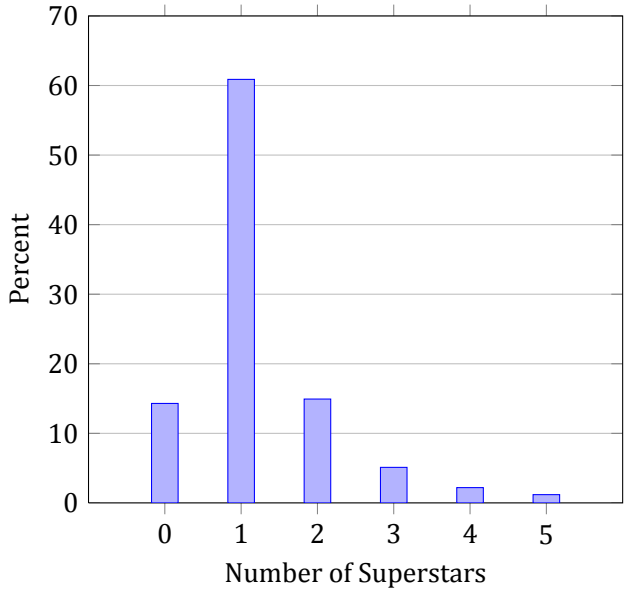


Figure 5: Number of groups in competition rooms.

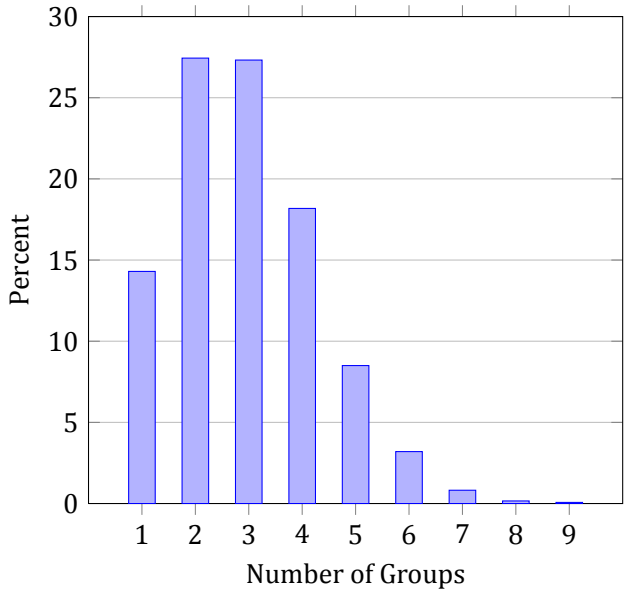


Figure 6: Response to competition across TopCoder ratings using a locally-weighted, kernel approach.

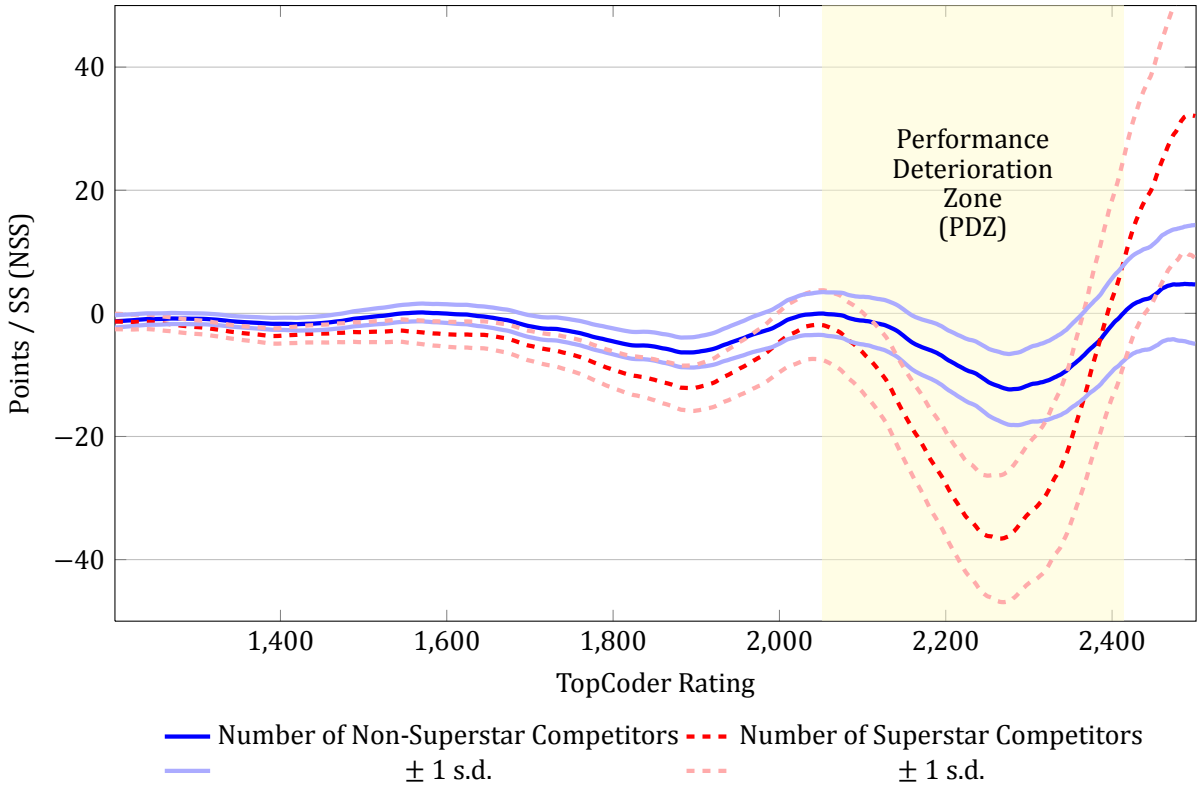


Figure 7: Response to competition for the three contest problems and challenge phase using kernel techniques.

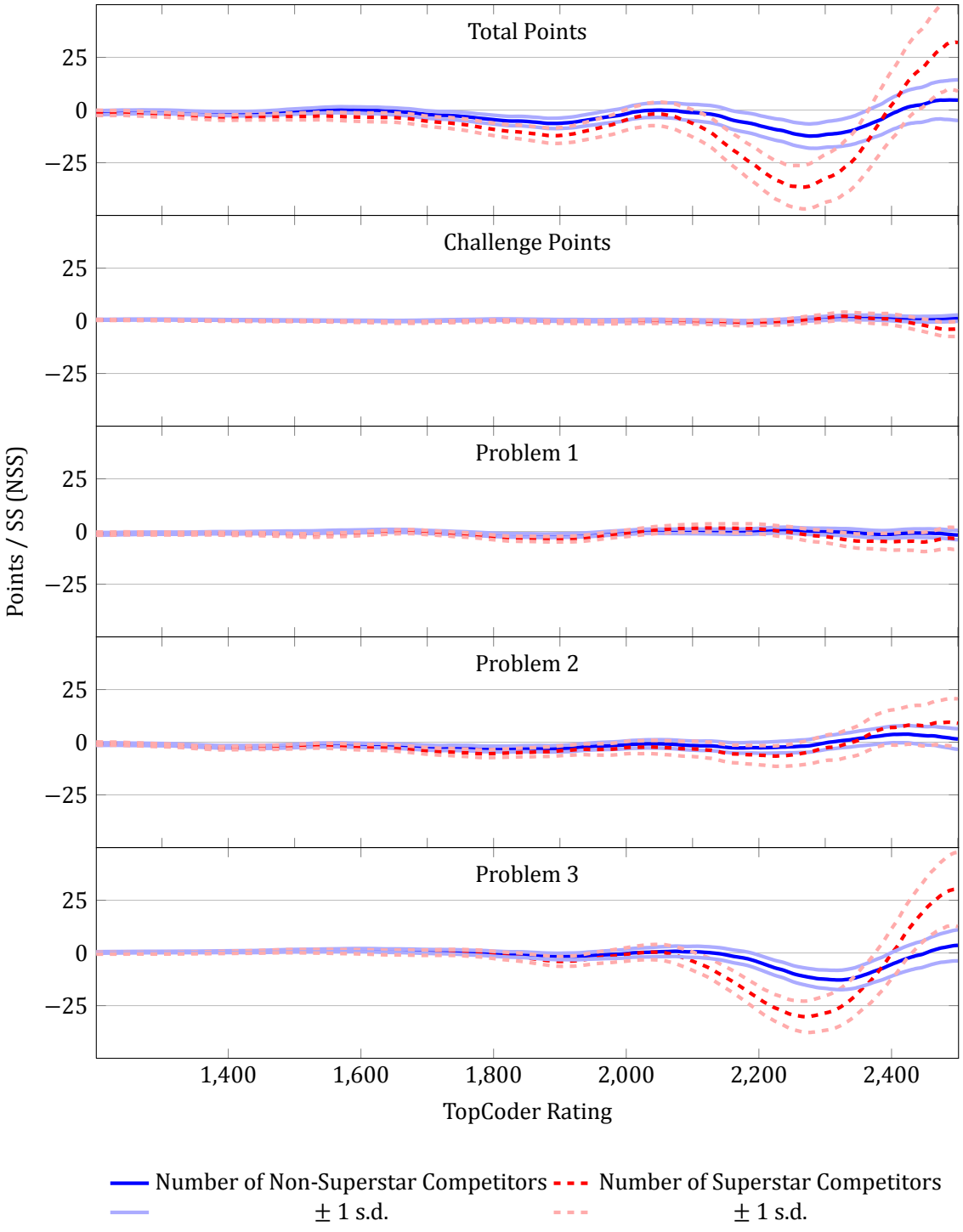
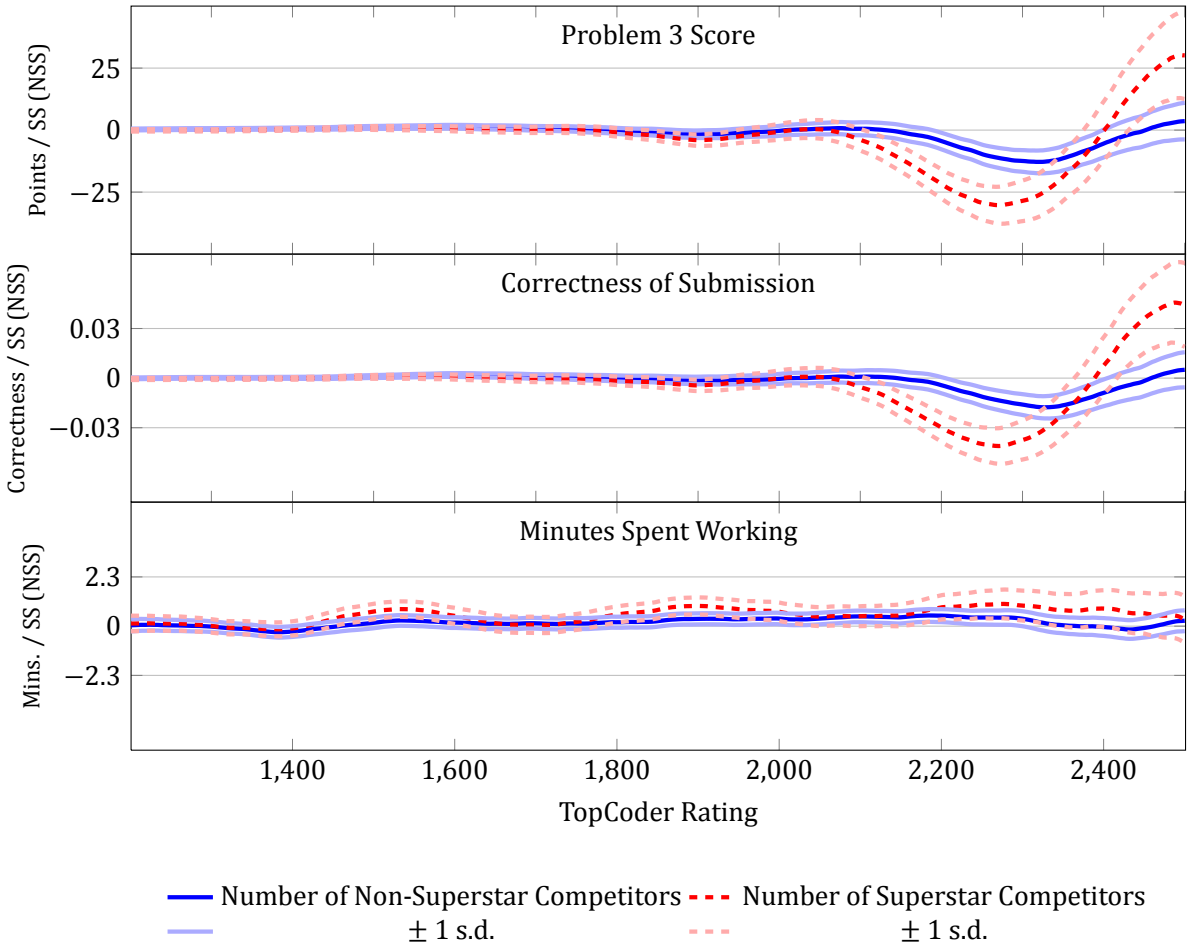


Figure 8: Response to competition on third problem: points, correctness, and time worked, using kernel techniques.



Tables

Table 1: The response of final points to number of competitors in the room.

Covariates	Final Points
Number in Room	-2.593^{***}
	(0.975)
TopCoder Rating	0.0731^{***}
	(0.00675)
1^{st} Problem Point Value	-0.724^{***}
	(0.0525)
2^{nd} Problem Point Value	-0.495^{***}
	(0.0284)
3^{rd} Problem Point Value	-0.146^{***}
	(0.0215)
Number in Contest	0.0245
	(0.0236)
Was Money Paid	10.54^{***}
	(2.698)
Contest Year	-5.843^{*}
	(3.229)
Constant	208.2^{***}
	(11.16)
Month Dummies	Yes
Day of Week Dummies	Yes
Observations	50,130
Number of Competitors	4,432

Competitor Fixed-effects, Standard errors in parentheses, *** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$

Table 2: The response of final points to the number of superstars and number of competitors in the room.

Covariates	Final Points
Number in Room	-2.606^{***}
	(0.974)
Number of Superstars	-2.670^{***}
	(0.845)
TC Rating	0.0727^{***}
	(0.00675)
1^{st} Problem Point Value	-0.724^{***}
	(0.0524)
2^{nd} Problem Point Value	-0.497^{***}
	(0.0284)
3^{rd} Problem Point Value	-0.147^{***}
	(0.0215)
Number in Contest	0.022
	(0.0236)
Was Money Paid	10.76^{***}
	(2.698)
Contest Year	-5.641^{*}
	(3.23)
Constant	208.0^{***}
	(11.16)
Month Dummies	Yes
Day of Week Dummies	Yes
Observations	50,130
Number of Competitors	4,432

Competitor Fixed-effects, Standard errors in parentheses, *** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$

Table 3: The response of final points to the number of superstars and non-superstars in the room.

Covariates	Final Points
Number of Non-Superstars	-2.606^{***}
	(0.974)
Number of Superstars	-5.277^{***}
	(1.293)
TC Rating	0.0727^{***}
	(0.00675)
1st Problem Point Value	-0.724^{***}
	(0.0524)
2nd Problem Point Value	-0.497^{***}
	(0.0284)
3rd Problem Point Value	-0.147^{***}
	(0.0215)
Number in Contest	0.0220
	(0.0236)
Was Money Paid	10.76^{***}
	(2.698)
Contest Year	-5.641^{*}
	(3.230)
Constant	207.6^{***}
	(11.14)
Month Dummies	Yes
Day of Week Dummies	Yes
Observations	50,139
Number of Competitors	4,432

Competitor Fixed-effects, Standard errors in parentheses,
*** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$

Table 4: Response to competition for the three contest problems and challenge phase.

Covariates	Final Points	1^{st} Problem Points	2^{nd} Problem Points	3^{rd} Problem Points	Challenge Points
Below PDZ × Number of Non-Superstars	-1.930^{*}	-0.760^{*}	-2.084^{***}	0.473	0.376^{**}
	(1.031)	(0.427)	(0.629)	(0.603)	(0.177)
PDZ × Number of Non-Superstars	-7.608^{**}	-1.194	-1.296	-5.141^{***}	0.111
	(3.144)	(1.302)	(1.918)	(1.84)	(0.54)
Above PDZ × Number of Non-Superstars	-9.775	-2.131	-2.743	-7.713^{**}	2.822^{**}
	(6.447)	(2.669)	(3.932)	(3.772)	(1.107)
Below PDZ × Number of Superstars	-4.388^{***}	-1.442^{***}	-2.764^{***}	-0.127	-0.0866
	(1.347)	(0.558)	(0.822)	(0.788)	(0.231)
PDZ × Number of Superstars	-19.83^{***}	-0.273	-3.376	-15.97^{***}	0.0542
	(5.615)	(2.324)	(3.425)	(3.285)	(0.964)
Above PDZ × Number of Superstars	31.00^{*}	-3.274	3.73	31.98^{***}	-1.385
	(16.57)	(6.86)	(10.11)	(9.695)	(2.845)
Constant	203.8^{***}	168.6^{***}	52.86^{***}	-18.75^{***}	0.9
	(11.17)	(4.623)	(6.812)	(6.533)	(1.917)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls;
Observations: 50,139; Number of competitors: 4,432; Competitor Fixed-effects, Standard errors in parentheses,
$^{***}\ p < 0.01$, $^{**}\ p < 0.05$, $^{*}\ p < 0.1$

Table 5: Response to competition for points on the third problem, with varying controls and specifications.

Covariates	OLS No Controls	OLS	Primary Specification	Contest FE	Competitor FE Experience
Below PDZ × Number of Non-Superstars	-0.198	0.767	0.473	1.323^{**}	0.497
	(0.568)	(0.568)	(0.603)	(0.585)	(0.603)
PDZ × Number of Non-Superstars	-5.843^{***}	-4.870^{***}	-5.141^{***}	-4.281^{**}	-5.153^{***}
	(1.825)	(1.795)	(1.84)	(1.74)	(1.839)
Above PDZ × Number of Non-Superstars	-7.157^{*}	-6.671^{*}	-7.713^{**}	-8.324^{**}	-7.654^{**}
	(3.719)	(3.649)	(3.772)	(3.514)	(3.771)
Below PDZ × Number of Superstars	-1.153	0.137	-0.127	1.122	-0.0935
	(0.746)	(0.739)	(0.788)	(0.745)	(0.788)
PDZ × Number of Superstars	-18.94^{***}	-16.40^{***}	-15.97^{***}	-15.50^{***}	-16.06^{***}
	(3.257)	(3.196)	(3.285)	(3.085)	(3.285)
Above PDZ × Number of Superstars	24.64^{***}	28.66^{***}	31.98^{***}	18.32^{**}	32.08^{***}
	(9.544)	(9.357)	(9.695)	(9.011)	(9.694)
Constant	18.15^{***}	-11.92^{*}	-18.75^{***}	28.60^{***}	-23.80^{***}
	(0.678)	(6.176)	(6.533)	(0.736)	(6.675)
Skill Control	No	No	Yes	Yes	Yes
Contest Controls	No	No	Yes	No	Yes
Competitor Fixed-Effects	No	Yes	Yes	No	Yes
Experience Control	No	No	No	No	Yes
Contest Fixed-Effects	No	No	No	Yes	No

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls;
Observations: 50,139; Number of competitors: 4,432;
Competitor Fixed-effects, Standard errors in parentheses, $^{***}\ p < 0.01$, $^{**}\ p < 0.05$, $^{*}\ p < 0.1$

Table 6: Response to competition with restricted numbers of superstars.

Covariates	All Rooms	Rooms with < 3 Superstars	Rooms with < 4 Superstars
Below PDZ × Number of Non-Superstars	0.473 (0.433)	0.0743 (0.514)	0.1 (0.398)
PDZ × Number of Non-Superstars	-5.141^{***} (0.0052)	-3.956^{*} (0.0709)	0.631 (0.811)
Above PDZ × Number of Non-Superstars	-7.713^{**} (0.0409)	11.59^{**} (0.0339)	11.14 (0.22)
Below PDZ × Number of Superstars	-0.127 (0.872)	-0.0348 (0.975)	-0.135 (0.876)
PDZ × Number of Superstars	-15.97^{***} (1.16E-06)	-28.98^{***} (2.14E-08)	-22.31^{***} (3.65E-06)
Above PDZ × Number of Superstars	31.98^{***} (0.000974)	73.71^{***} (-4.17E-08)	42.75^{**} (0.0116)
Constant	-26.75^{**} (0.0416)	-18.46^{***} (0.00558)	-18.18^{***} (0.00656)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls;
Observations: 50,139; Number of competitors: 4,432;
Competitor Fixed-effects, Standard errors in parentheses, $^{***}\ p < 0.01$, $^{**}\ p < 0.05$, $^{*}\ p < 0.1$

Table 7: Response to competition on third problem: likelihood of opening the problem and time spent working.

Covariates	3^{rd} Not Opened	Min. Worked on 3^{rd} Submissions	Min. Worked on Correct 3^{rd} Submissions
PDZ × Number of Non-Superstars	-0.00108 (0.00554)	0.643^{**} (0.322)	0.385 (0.435)
Above PDZ × Number of Non-Superstars	0.00391 (0.0114)	0.153 (0.519)	-0.0693 (0.658)
PDZ × Number of Superstars	-0.0105 (0.00989)	0.969 (0.598)	1.344^{*} (0.809)
Above PDZ × Number of Superstars	0.00357 (0.0292)	-0.0948 (1.247)	-0.00679 (1.572)
Constant	0.297^{***} (0.0196)	33.37^{***} (2.103)	33.34^{***} (2.088)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls;
Observations: 50,139 (column 1), 6,174 (columns 2 & 3);
Number of competitors: 4,432 (column 1), 1,461 (columns 2 & 3);
Competitor Fixed-effects, Standard errors in parentheses, $^{***}\ p < 0.01$, $^{**}\ p < 0.05$, $^{*}\ p < 0.1$

Table 8: Response to competition on third problem: likelihood of errors in submissions.

Covariates	3rd Incorrect cond. on submitting
PDZ × Number of Non-Superstars	0.0209***
	(0.00487)
Above PDZ × Number of Non-Superstars	0.0116
	(0.00792)
PDZ × Number of Superstars	0.0495***
	(0.00903)
Above PDZ × Number of Superstars	-0.0118
	(0.0192)
Constant	1.039***
	(0.00995)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls; Observations: 50,139; Number of competitors: 4,432; Competitor Fixed-effects, Standard errors in parentheses, *** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$

Table 9: Likelihood of competitors, who opened but did not submit the third problem issuing a challenge.

Covariates	Issued Challenge
PDZ × Number of Non-Superstars	-0.0116
	(0.0255)
Above PDZ × Number of Non-Superstars	-0.0523
	(0.0962)
PDZ × Number of Superstars	0.0627
	(0.0545)
Above PDZ × Number of Superstars	0.542*
	(0.323)
Constant	0.279***
	(0.0217)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls; Observations: 50,139; Number of competitors: 4,432; Competitor Fixed-effects, Standard errors in parentheses, *** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$

Table 10: Likelihood of incorrect submissions for the third problem controlling for minutes spent working.

Covariates	3rd Incorrect cond. on submitting
PDZ × Number of Non-Superstars	0.0149***
	(0.00486)
Above PDZ × Number of Non-Superstars	0.00930
	(0.00788)
PDZ × Number of Superstars	0.0408***
	(0.00900)
Above PDZ × Number of Superstars	-0.0162
	(0.0191)
PDZ × Min. Spent Working	0.00709***
	(0.000399)
Above PDZ × Min. Spent Working	0.00662
	(0.000593)
Constant	1.037***
	(0.00990)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls; Observations: 50,139; Number of competitors: 4,432; Competitor Fixed-effects, Standard errors in parentheses, *** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$

Table 11: Descriptive statistics

Variable	Mean	Std. Dev.	Min	Max
TopCoder Rating	1603.82	329.24	1200	3375
Number of Contests (prior experience)	35.63	35.91	1	273
Final Points in Contest	246.81	256.67	-400	1972.53
Final Points on 1^{st} Problem*	129.28	99.1	0	308
Final Points on 2^{nd} Problem*	84.44	145.84	0	643
Final Points on 3^{rd} Problem*	29.99	132.93	0	990
Final Points on 1^{st} Problem (cond. on submitting)	142.42	94.6	0	308
Final Points on 2^{nd} Problem (cond. on submitting)	162.76	168.08	0	643
Final Points on 3^{rd} Problem (cond. on submitting)	243.58	302.48	0	990
Challenge Points	2.86	37.56	-500	700
Minutes Worked on 3^{rd} Problem	3.92	11.46	0	74.88
3^{rd} Problem Incorrect*	0.95	0.22	0	1
Minutes Worked on 3^{rd} Problem (cond. on submitting)	31.83	13.31	0.15	74.88
3^{rd} Problem Incorrect (cond. on submitting)	0.58	0.49	0	1
Number of Competitors in Contest	352.1	127.82	113	619
Number of Competitors in Room	18.61	1.06	13	20
Total Points Available in Contest**	1753.89	53.98	1550	1950
Point Value of First Problem	256.63	20.98	200	375
Point Value of Second Problem	511.61	41.39	400	675
Point Value of Third Problem	985.65	51.67	750	1200
Number of Superstars per room	1.31	1.21	0	16
Number of Non-superstars per room	17.3	1.61	2	20
Number of Groups in room	2.93	1.36	1	9
Dollars Paid Out in Prizes	1323.4	2205.9	0	5032
Number of Rooms per Contest	16.2	6.65	7	34
Number of Blue Coders per Room	8.17	3.23	0	20
Number of Yellow Coders per Room	8.34	2.52	0	16
Number of Red Coders per Room	2.1	1.8	0	9
Number of Below PDZ Coders per Room	15.46	2.53	7	20
Number of PDZ Coders per Room	1.99	1.57	0	8
Number of Above PDZ Coders per Room	1.15	1.28	0	8

* *Includes coders who did not make submissions and therefore earned 0 points*
** *Calculated as the sum of the point values for the three problems, exclusive of challenge points*

Table 12: Response to competition using dummy variables to indicate number of superstars.

Covariates	Final Points
Below PDZ × Number of Non-Superstars	-1.46
	(0.981)
PDZ × Number of Non-Superstars	-7.521^{**}
	(3.14)
Above PDZ × Number of Non-Superstars	-9.351
	(6.444)
Below PDZ × 1 Superstar	-7.420^{**}
	(3.344)
Below PDZ × 2 Superstars	-8.237^{*}
	(4.471)
Below PDZ × 3 Superstars	-9.513
	(6.22)
Below PDZ × 4 Superstars	-16.40^{*}
	(8.421)
Below PDZ × 5+ Superstars	-30.30^{***}
	(9.43)
PDZ × 1 Superstar	-1.599
	(12.73)
PDZ × 2 Superstars	-54.62^{***}
	(15.74)
PDZ × 3 Superstars	-61.91^{***}
	(23.43)
PDZ × 4 Superstars	26.22
	(38.08)
PDZ × 5+ Superstars	-38.97
	(55.47)
Above PDZ × 1 Superstar	15.9
	(45.02)
Above PDZ × 2 Superstars	71.68
	(49.4)
Above PDZ × 3 Superstars	-3.218
	(73.09)
Above PDZ × 4 Superstars	No Obs.
	-
Above PDZ × 5+ Superstars	No Obs.
	-
Constant	232.7^{***}
	(11.7)

Includes: PDZ Indicators (Below Omitted), Skill Control, and Contest Controls; Observations: 50,139; Number of competitors: 4,432; Competitor Fixed-effects, Standard errors in parentheses, $^{***}\ p < 0.01$, $^{**}\ p < 0.05$, $^{*}\ p < 0.1$