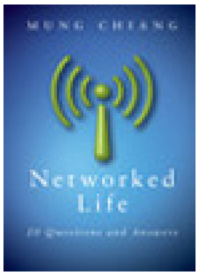

Networked Life

20 Questions and Answers

Mung Chiang

Book DOI: http://dx.doi.org/10.1017/CBO9781139176200

Online ISBN: 9781139176200

Hardback ISBN: 9781107024946

Chapter

1 - What makes CDMA work for my smartphone? pp. 1-24

Chapter DOI: http://dx.doi.org/10.1017/CBO9781139176200.003

Cambridge University Press

1 What makes CDMA work for my smartphone?

1.1 A Short Answer

Take a look at your iPhone, Android phone, or a smartphone running on some other operating system. It embodies a remarkable story of technology innovations. The rise of wireless networks, the Internet, and the web over the last five decades, coupled with advances in chip design, touchscreen material, battery packaging, software systems, business models... led to this amazing device you are holding in your hand. It symbolizes our age of networked life.

These phones have become the mobile, lightweight, smart centers of focus in our lives. They are used not just for voice calls, but also for **data applications**: texting, emailing, browsing the web, streaming videos, downloading books, uploading photos, playing games, or video-conferencing friends. The throughputs of these applications are measured in bits per second (bps). These data fly through a **cellular network** and the **Internet**. The cellular network in turn consists of the radio air-interface and the core network. We focus on the air-interface part in this chapter, and turn to the cellular core network in Chapter 19.

Terrestrial wireless communication started back in the 1940s, and cellular networks have gone through generations of evolution since the 1970s, moving into what we hear as 4G these days. Back in the 1980s, some estimated that there would be 1 million cellular users in the USA by 2000. That turned out to be one of those way-off under-estimates that did not even get close to the actual impact of networking technologies.

Over more than three decades of evolution, a fundamental concept of cellular architecture has remained essentially the same. The entire space of deployment is divided into smaller regions called **cells**, which are often represented by hexagons as in Figure 1.1, thus the name cellular networks and cell phones. There is one **base station** (BS) in each cell, connected on the one side to switches in the core network, and on the other side to the **mobile stations** (MSs) assigned to this cell. An MS could be a smart phone, a tablet, a laptop with a dongle, or any device with antennas that can transmit and receive in the right frequencies following a cellular network standard. There are a few other names for them, for example, sometimes an MS is also called a User Equipment (UE) and a BS called a Node B (NB) or an evolved Node B (eNB).

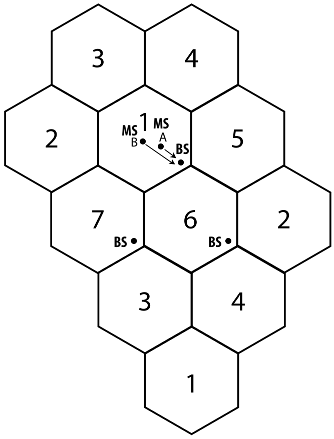

Figure 1.1 Part of a typical cellular network with a frequency reuse of 7. Each cell is a hexagon with a base station (BS) and multiple mobile stations (MSs). Only a few of them are drawn in the figure. Each BS has three directional antennas, each of which covers a 120-degree sector. Some mobile stations, like MS A, are close to the base station with strong channels to the BS. Others, like MS B, are on the cell edge with weak channels. Attenuation enables frequency reuse, but the variability of and the inability to control attenuation pose challenges to wireless cellular network design.

We see a clear hierarchy, a fixed infrastructure, and one-hop radio links in cellular networks. This is in contrast to other types of wireless networks. Moreover, the deployment of base stations is based on careful radio engineering and tightly controlled by a wireless provider, in contrast to WiFi networks in Chapter 18.

Why do we divide the space into smaller regions? Because the wireless spectrum is scarce and radio signals weaken over space.

Transmitting signals over the air means emitting energy over different parts of the electromagnetic **spectrum**. Certain regions of the spectrum are allocated by different countries to cellular communications, just like other parts of the spectrum are allocated to AM and FM radio. For example, the 900 MHz range is allocated for the most popular 2G standard called GSM, and in Europe the 1.95 GHz range and 2.15 GHz range are allocated for UMTS, a version of the 3G standard. Some part of the spectrum is unlicensed, like in WiFi, as we will see in Chapter 18. Other parts are licensed, like those for cellular networks, and a wireless service provider needs to purchase these limited resources with hefty prices. The spectrum for cellular networks is further divided into chunks, since it is often easier for transmitters and receivers to work with narrower frequency bands, e.g., on the order of 10 MHz in 3G.

The signals sent in the air become weaker as they travel over longer distances. The amount of this attenuation is often proportional to the square, or even the fourth power, of the distance traversed. So, in a typical cellular network, the signals become too weak to be accurately detected after a couple of miles. At

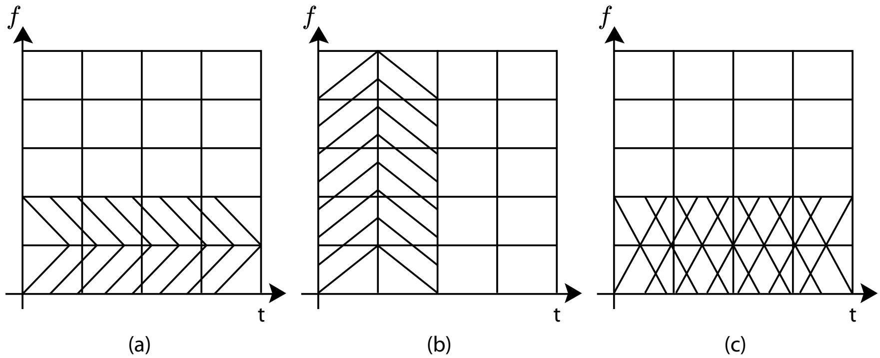

Figure 1.2 Part of a time–frequency grid is shown in each graph. For visual clarity, we only show two slices of resources being used. (a) FDMA and (b) TDMA are dedicated resource allocation: each frequency band or timeslot is given to a user. In contrast, (c) CDMA is shared resource allocation: each time–frequency bin is shared by multiple users, all transmitting and receiving over the same frequency band and at the same time. These users are differentiated by signal processing. Power control also helps differentiate their signals.

first glance, this may sound like bad news. But it also means that the frequency band used by base station A can be reused by another base station B sufficiently far away from A. All we need to do is tessellate the frequency bands, as illustrated in Figure 1.1, so that no two cells share the same frequency band if they are too close. In Figure 1.1, we say that there is a **frequency reuse factor** of 7, since we need that many frequency bands in order to avoid having two close cells sharing the same frequency band. **Cellular architecture** enables the network to scale up over space. We will visit several other ways to scale up a network later.

Now, how can the users in the *same* cell share the same frequency band? There are two main approaches: orthogonal and non-orthogonal allocation of resources.

Frequency is clearly one type of resource, and time is another. In **orthogonal allocation**, each user is given a small band of frequency in Frequency-Division Multiple Access (**FDMA**), or a timeslot in Time-Division Multiple Access (**TDMA**). Each user's allocation is distinct from others, as shown in Figure 1.2(a) and (b). This often leads to an inefficient use of resources. We will see in later chapters a recurring theme: a dedicated assignment of resources to users becomes inefficient when users come and go frequently.

The alternative, **non-orthogonal allocation**, allows all users to transmit at the same time over the same frequency band, as in Code-Division Multiple Access. **CDMA** went through many ups and downs with technology adoption from 1989 to 1995, but is now found in all the 3G cellular standards as part of the design. In CDMA's first standard, IS-95 in the 2G family, the same frequency band is reused in all the cells, as illustrated in Figure 1.2(c). But how can we distinguish the users if their signals overlap with each other?

Think of a cocktail party with many pairs of people trying to carry out individual conversations. If each pair takes turns in communicating, and only one person gets to talk during each timeslot, we have a TDMA system. If all pairs can communicate at the same time, and each uses a different language to avoid confusion, we have a CDMA system. But there are not enough languages whose pronunciations do not cause confusion, and human ears are not that good at decoding, so interference is still an issue.

How about controlling each person's volume? Each transmitter adjusts the volume of its voice according to the relative distances among the speakers and listeners. In a real cocktail party, unless there is some politeness protocol or it hurts people's vocal chord to raise their voice, we end up in a situation where everyone is shouting and yet most people cannot hear well. Transmit power control should mitigate this problem.

The core idea behind the CDMA standards follows our intuition about the cocktail party. First, the transmitter multiplies the digital signals by a sequence of 1s and minus 1s, a sequence we call the **spreading code**. The receiver multiplies the received bits by the same spreading code to recover the original signals. This is straightforward to see: 1×1 is 1, and -1×-1 is also 1. What is non-trivial is that a family of spreading codes can be designed such that only *one* spreading code, the original one used by the transmitter, can recover the signals. If you use any other spreading code in this family, you will get noise-like, meaningless bits. We call this a family of **orthogonal codes**. Users are still separated by orthogonalization, just along the "code dimension" as opposed to the more intuitive "time dimension" and "frequency dimension." This procedure is called direct sequence **spread spectrum**, one of the standard ways to enable CDMA.

However, there may not be enough orthogonal spreading codes for all the mobile stations. Families of orthogonal codes are limited in their sizes. Furthermore, a slight shift on the time axis can scramble the recovered bits at the receiver. We need the clocks on all the devices to be synchronized. But this is infeasible for the **uplink**, where mobiles talk to the base station: MSs cannot easily coordinate their clocks. It is difficult even in the **downlink**, where the base station talks to the mobiles: the BS has a single clock but the wireless channel distorts the bits. Either way, we do not have perfectly orthogonal spreading codes, even though these imperfect codes still provide significant "coding gain" in differentiating the signals.

We need an alternative mechanism to differentiate the users and to tackle the **interference** problem. Wireless signals are just energy propagating in the air, and one user's signal is every other user's interference. Interference, together with the attenuation of signals over distance and the fading of signals along multiple paths, are the the top three issues we have to address in wireless channels. Interference is an example of **negative externality** that we will encounter many times in this book, together with ways to "internalize" it by designing the right mechanism.

Here is an example of significant interference. As shown in Figure 1.1, a user standing right next to the BS can easily overwhelm another user far away at the edge of the cell. This is the classic **near-far problem** in CDMA networks. It was solved in the IS-95 standard by Qualcomm in 1989. This solution has been one of the cornerstones in realizing the potential of CDMA since then.

Qualcomm's solution to the near-far problem is simple and effective. The receiver infers the channel quality and sends that back to the transmitter as feedback. Consider an uplink transmission: multiple MSs trying to send signals to the BS in a particular cell. The BS can estimate the channel quality from each MS to itself, e.g., by looking at the ratio of the received signal power to the transmitted power, the latter being pre-configured to some value during the channel-estimation timeslot. Then, the BS inverts the channel quality and sends that value, on some feedback control channel, back to the MSs, telling them that these are the gain parameters they should use in setting their transmit powers. In this way, all the received signal strengths will be made equal. This is the basic MS **transmit power control** algorithm in CDMA.

But what if equalization of the received signal powers is *not* the right goal? For voice calls, the typical application on cell phones in 2G networks in the 1990s, there is often a target value of the received signal quality that each call needs to achieve. This signal quality factor is called the Signal to Interference Ratio (**SIR**). It is the ratio between the received signal strength and the sum strength of all the interference (plus the receiver noise strength). Of course, it is easy to raise the SIR for just one user: just increase its transmitter's power. But that translates into higher interference for everyone else, which further leads to higher transmit powers being used by them if they also want to maintain or improve their SIRs. This positive feedback escalates into a transmit power "arms race" until each user is transmitting at the maximum power. That would not be a desirable state to operate in.

If each user fixes a reasonable target SIR, can we do better than this "arms race" through a more intelligent power control? Here, "being reasonable" means that the SIRs targeted by all the users in a cell are mutually compatible; they *can* be simultaneously achieved by some configuration of transmit powers at the MSs.

The answer is yes. In 1992-1993, a sequence of research results developed the basic version of **Distributed Power Control** (DPC), a fully **distributed algorithm**. We will discuss later what we mean by "distributed" and "fully distributed." For now, it suffices to say that, in DPC, each pair of transmitter (e.g., an MS) and receiver (e.g., the BS) does not need to know the transmit power or channel quality of any other pair. At each timeslot, all it needs to know is the actual SIR it currently achieves at the receiver. Then, by taking the ratio between the fixed, target SIR and the variable, actual SIR value measured for this timeslot, and multiplying the current transmit power by that ratio, we get the transmit power for the next timeslot. This update happens simultaneously at each pair of transmitter and receiver.

This simple method is an **iterative algorithm**; the updates continue from one timeslot to the next, unlike with the one-shot, received-power-equalization algorithm. But it is still simple, and when the target SIRs can be simultaneously achieved, it has been proven to **converge**: the iterative procedure will stop over time. When it stops, it stops at the right solution: a power-minimal configuration of transmit powers that achieves the target SIRs for all. DPC converges quite fast, approaching the right power levels with an error that decays as a geometric series. DPC can even be carried out asynchronously: each radio has a different clock and therefore different definitions of what timeslot it is now.

Of course, in real systems the timeslots are indeed *asynchronous* and power levels are *discrete*. Asynchronous and quantized versions of DPC have been implemented in all the CDMA standards in 3G networks. Some standards run power control 1500 times every second, while others run 800 times a second. Some discretize power levels to 0.1 dB, while others between 0.2 and 0.5 dB. Without CDMA, our cellular networks today could not work as efficiently. Without power control algorithms (and the associated handoff method to support user mobility), CDMA could not function properly. In Chapter 19, we will discuss a 4G standard called LTE. It uses a technology called OFDM instead of CDMA, but power control is still employed for interference reduction and for energy management.

Later, in Chapter 18, we will discuss some of the latest ideas that help further push the data rates in new wireless network standards, ranging from splitting, shrinking, and adjusting the cells to overlaying small cells on top of large ones for traffic offloading, and from leveraging multiple antennas and tilting their positions to "chopping up" the frequency bands for more efficient signal processing.

1.2 A Long Answer

1.2.1 Distributed power control

Before we proceed to a general discussion of the Distributed Power Control (DPC) algorithm, we must first define some symbols.

Consider N pairs of transmitters and receivers. Each pair forms a (logical) link, indexed by i. The transmit power of the transmitter of link i is p_i, some positive number, usually capped at a maximum value: $p_i \leq p_{max}$ (although we will not consider the effect of this cap in the analysis of the algorithm). The transmitted power impacts both the received power at the intended receiver and the received interference at the receivers of all other pairs.

Now, consider the channel from the transmitter of link (i.e., transmitter–receiver pair) j to the receiver of link i, and denote the **channel gain** by G_{ij}. So G_{ii} is the direct channel gain; the bigger the better, since it is the channel for the intended transmission for the transmitter–receiver pair of link i. All the other $\{G_{ij}\}$, for j not equal to i, are gains for interference channels, so the smaller the better. We call these channel "gains", but actually they are less than 1, so maybe a better term is channel "loss."

This notation is visualized in Figure 1.3 for a simple case of two MSs talking to a BS, which can be thought of as two different (logically separated) receivers physically located together.

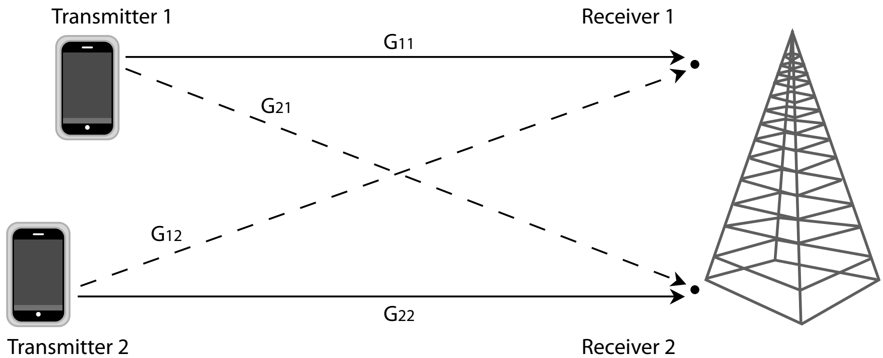

Figure 1.3 Uplink interference between two mobile stations at the base station. We can think of the base station as two (logically separated) receivers collocated. G_{11} and G_{22} are direct channel gains, the bigger the better. G_{12} and G_{21} are interference channel gains, the smaller the better.

Each G_{ij} is determined by two main factors: (1) location of the transmitter and receiver and (2) the quality of the channel in between. G_{ii} is also enhanced by the CDMA spreading codes that help the intended receivers decode more accurately.

The received power of the intended transmission at the receiver is therefore $G_{ii}p_i$. What about the interference? It is the sum of $G_{ij}p_j$ over all transmitters j (other than the intended one i): $\sum_{j \neq i} G_{ij}p_j$. There is also noise n_i in the receiver electronics for each receiver i. So we can write the SIR, a unit-less ratio, at the receiver of logical link i as

$$\mathrm{SIR}_i = \frac{G_{ii}p_i}{\sum_{j \neq i} G_{ij}p_j + n_i}. \tag{1.1}$$

For proper decoding of the packets, the receiver needs to maintain a target level of SIR. We will denote that as γ_i for link i, and we want $\mathrm{SIR}_i \geq \gamma_i$ for all i. Clearly, increasing p_1 raises the SIR for receiver 1 but lowers the SIR for all other receivers.

As in a typical algorithm we will encounter throughout this book, we assume that time is divided into discrete slots, each indexed by $[t]$. At each timeslot t, the receiver on link i can measure the received SIR readily, and feeds back that number, $\mathrm{SIR}_i[t]$, to the transmitter.

The DPC algorithm can be described through a simple equation: each transmitter simply multiplies the current power level $p_i[t]$ by the ratio between the target SIR, γ_i, and the current measured $\mathrm{SIR}_i[t]$, to obtain the power level to use in the next timeslot:

$$p_i[t+1] = \frac{\gamma_i}{\mathrm{SIR}_i[t]} p_i[t], \quad \text{for each } i. \tag{1.2}$$

We will use the symbols $\forall i$ later to say "for each i."

We see that each receiver i needs to measure only its own SIR at each iteration, and each transmitter only needs to remember its own target SIR. There is no need for passing any control message around, like telling other users what power level it is using. Simple in *communication*, it is a *very* distributed algorithm, and we will later encounter many types of distributed algorithms in various kinds of networks.

This algorithm is also simple in its *computation*: just one division and one multiplication. And it is simple in its parameter *configuration*: there are actually no parameters in the algorithm that need to be tuned, unlike in quite a few other algorithms in later chapters. Simplicity in communication, computation, and configuration is a key reason why certain algorithms are widely adopted in practice.

Intuitively, this algorithm makes sense. First, when the iterations stop because no one's power is changing any more, i.e., when we have convergence to an **equilibrium**, we can see that $\text{SIR}_i = \gamma_i$ for all i.

Second, there is hope that the algorithm will actually converge, given the direction in which the power levels are moving. The transmit power moves up when the received SIR is below the target, and moves down when it is above the target. *Proving* that convergence will happen is not as easy. As one transmitter changes its power, the other transmitters are doing the same, and it is unclear what the next timeslot's SIR values will be. In fact, this algorithm does *not* converge if too many γ_i are too large, i.e., when too many users request large SIRs as their targets.

Third, if satisfying the target SIRs is the only criterion, there are many transmit power configurations that can do that. If $p_1 = p_2 = 1$ mW achieves these two users' target SIRs, $p_1 = p_2 = 10$ mW will do so too. We would like to pick the configuration that uses the least amount of power; we want a power-minimal solution. And the algorithm above seems to be lowering the power when a high power is unnecessary.

DPC illustrates a recurring theme in this book. We will see in almost every chapter that individual's behaviors driven by self-interest can often aggregate into a fair and efficient state globally across all users, especially when there are proper feedback signals. In contrast, either a centralized control or purely random individual actions would have imposed significant downsides.

1.2.2 DPC as an optimization solution

In general, "will it converge?" and "will it converge to the right solution?" are the top two questions that we would like to address in the design of all iterative algorithms. Of course, what "the right solution" means will depend on the definition of optimality. In this case, power-minimal transmit powers that achieve the target SIRs for all users are the "right solution." Power minimization is the **objective** and achieving target SIRs for all users is the **constraint**.

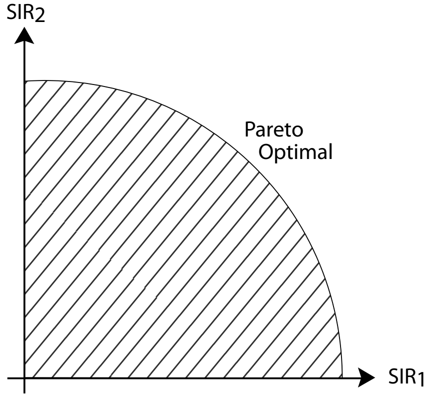

Figure 1.4 An illustration of the SIR feasibility region. It is a constraint set for power control optimization, and visualizes the competition among users. Every point strictly inside the shaded region is a feasible vector of target SIRs. Every point outside is infeasible. And every point on the boundary of the curve is Pareto optimal: you cannot increase one user's SIR without reducing another user's SIR.

In this case, there are many ways to address these questions, for example, using machineries from optimization theory or game theory. Either way, we can show that, under the condition that the target SIR values are indeed **achievable** at all (i.e., there are some values of the transmit powers that can achieve the target SIRs for all users), DPC will converge, and converge to the right solution.

We can illustrate a typical set of feasible SIRs in the SIR feasibility region shown in Figure 1.4. Clearly, we want to operate on the boundary of this region, and each point on the boundary is called **Pareto optimal**. Along this boundary, one user's higher SIR can be achieved, but at the expense of a lower SIR for another user. This highlights another recurrent theme in this book: the need to tackle tradeoffs among competing users and across different design objectives, and the importance of providing incentives for people to react to. There is no free lunch; and we have to balance the benefits with the costs.

It turns out that DPC also solves a global optimization problem for the network. Here, "global" means that the interests of *all* the users are incorporated. In this case, it is the sum of transmit powers that is minimized, and every user's target SIR must be met.

Once we have an objective function and a set of constraints, and we have defined which quantities are variables and which are constants, an **optimization** problem is formulated. In this case, the transmit powers are the variables. The achieved SIRs are also variables, but are derived from the powers. All the other quantities are the constants: they are not degrees of freedom under your control.

If the variable vector $\mathbf{x}_0$ satisfies all the constraints, it is called a **feasible solution**. If an optimization problem's constraints are not mutually compatible, it is called **infeasible**. If an $\mathbf{x}^*$ is both feasible and better than any other feasible solution, i.e., gives the smallest objective function value for a minimization problem (or the largest objective function value for a maximization problem), it is called an **optimal** solution. An optimal solution might not exist, e.g., minimize $1/x$ for $x \in \mathcal{R}$. And optimal solutions may not be unique either.

Here is the optimization problem of varying transmit power to satisfy fixed target SIR constraints and then minimize the total power:

$$\begin{array}{ll} \text{minimize} & \sum_i p_i \\ \text{subject to} & \mathsf{SIR}_i(\mathbf{p}) \geq \gamma_i, \;\; \forall i \\ \text{variables} & \mathbf{p}. \end{array} \tag{1.3}$$

Problem (1.3) looks complicated if we substitute the definition (1.1) of the SIR as a function of the whole vector $\mathbf{p}$:

$$\begin{array}{ll} \text{minimize} & \sum_i p_i \\ \text{subject to} & \frac{G_{ii}p_i}{\sum_{j\neq i} G_{ij}p_j + n_i} \geq \gamma_i, \;\; \forall i \\ \text{variables} & \mathbf{p}. \end{array}$$

But it can be simplified through a different representation. We can easily rewrite it as a **linear programming** problem: minimizing a linear function of the variables subject to linear constraints of the variables:

$$\begin{array}{ll} \text{minimize} & \sum_i p_i \\ \text{subject to} & G_{ii}p_i - \gamma_i(\sum_{j\neq i} G_{ij}p_j + n_i) \geq 0, \;\; \forall i \\ \text{variables} & \mathbf{p}. \end{array}$$

Linear programming problems are easy optimization problems. More generally, convex optimization (to be introduced in Chapter 4) is easy; easy in theory in terms of complexity and easy in practice with fast solution software. In the Advanced Material, we will derive DPC as the solution to problem (1.3).

We will be formulating and solving optimization problems many times in future chapters. Generally, solving a global optimization via local actions by each user requires explicit signaling. But in this case, it turns out that the selfish behavior of users in their own power minimization also solves the global, constrained optimization problem; their interests are correctly aligned already. This is more of an exception than the norm.

1.2.3 DPC as a game

Power control is a competition. One user's received power is another's interference. Each player searches for the right "move" (or, in this case, the right transmit power) so that its "payoff" is optimized (in this case, the transmit power is the smallest possible while providing the user with its target SIR γ_i). We also hope that the whole network reaches some desirable equilibrium as each player strategizes. The concepts of "players," "move," "payoff," and "equilibrium" can be defined in a precise and useful way.

We can model *competition* as a **game**. The word "game" here carries a technical meaning. The study of games is a branch of mathematics called game theory. If the competition is among human beings, a game might actually correspond to people's strategies. If it is among devices, as in this case among radios, a game is more like an angle of interpretation and a tool for analysis. It turns out that *cooperation* can also be modeled in the language of game theory, as we will show in Chapter 6.

In the formal definition, a game is specified by three elements:

1. a set of **players** $\{1, 2, \ldots, N\}$,
2. a **strategy space** A_i for each player i, and
3. a **payoff function**, or utility function, U_i for each player to maximize (or a **cost function** to minimize). Function U_i maps each combination of all players' strategies to a real number, the payoff (or cost), to player i.

	`Not Confess`	`Confess`
`Not Confess`	$(-1, -1)$	$(-5, 0)$
`Confess`	$(0, -5)$	$(-3, -3)$

Table 1.1 Prisoner's dilemma. This is a famous game in which there is a unique and undesirable Nash equilibrium. Player A's two strategies are the two rows. Player B's two strategies are the two columns. The values in the table represent the payoffs to the two players in each scenario.

Now consider the two-player game in Table 1.1. This is the famous **prisoner's dilemma** game, which we will also encounter later in voting paradoxes, tragedy of the commons, and P2P file sharing. The two players are two prisoners. Player A's strategies are shown in rows and player B's in columns. Each entry in the 2×2 table has two numbers, (x, y), where x is the payoff to A and y that to B if the two players pick the corresponding strategies. As you would expect from the coupling between the players, each payoff value is determined jointly by the strategies of both players. For example, the payoff function maps `(Not Confess, Not Confess)` to -1 for both players A and B. These payoffs are negative because they are the numbers of years the two prisoners are going to serve in prison. If one confesses but the other does not, the one who confesses gets a deal to walk away free and the other one is heavily penalized. If both confess, both serve three years. If neither confesses, only a lesser conviction can be pursued and both serve only one year. Both players know this table, but they cannot communicate with each other.

If player A chooses the strategy `Not Confess`, player B should choose the strategy `Confess`, since $0 > -1$. This is called the **best response strategy** by player B, in response to player A choosing the strategy `Not Confess`.

If player A chooses the strategy `Confess`, player B's best response strategy is still `Confess`, since $-3 > -5$. When the best response strategy of a player is the *same* no matter what strategy the other player chooses, we call that a **dominant strategy**. It might not exist. But, when it does, a player will obviously pick a dominant strategy.

In this case, `Confess` is the dominant strategy for player B. By symmetry, it is also the dominant strategy for player A. So both players will pick `Confess`, and `(Confess, Confess)` is an **equilibrium** for the game. This is a slightly different definition of "equilibrium" from what we saw before, where equilibrium means an update equation reaches a fixed point.

Clearly, this equilibrium is undesirable: `(Not Confess, Not Confess)` gives a higher payoff value to both players: -1 instead of -3. But the two prisoners could not have coordinated to achieve `(Not Confess, Not Confess)`. An equilibrium might not be **socially optimal**, i.e., a set of strategies maximizing the sum of payoffs $\sum_i U_i$ of all the players. It might not even be Pareto optimal, i.e., a set of strategies such that no player's payoff can be increased without hurting another player's payoff.

	`Action Movie`	`Romance Movie`
`Action Movie`	(2, 1)	(0, 0)
`Romance Movie`	(0, 0)	(1, 2)

Table 1.2 Coordination Game. In this game, there are two Nash equilibria. Neither player has an incentive to unilaterally change strategy in either equilibrium. Rows are your strategies and columns your friend's.

Consider a different game in Table 1.2. This is a typical game model for **coordination**, a task we will see many times in future chapters. You and your friend are trying to coordinate which movie to watch together. If you disagree, you will not go to any movie and the payoff is zero for both of you. If you agree, each will get some positive payoff but the values are different, as you prefer the action movie and your friend prefers the romance movie. (By the way, we will try to understand how to predict a person's preference for different types of movies in Chapter 4.)

In this game, there is no dominant strategy, but it so happens that there are pairs of best response strategies that match each other. If my best response to my friend picking strategy a is strategy b, and my friend's best response to my picking strategy b is strategy a, then the (a, b) pair "matches."

In the coordination game, `(Action, Action)` is such a pair, and `(Romance, Romance)` is another pair. For both pairs, neither player has an incentive to *unilaterally* move away from its choice in this pair of strategies. If both move at the same time, they could both benefit, but neither wants to do that *alone*. This creates an equilibrium in strategic thinking: I will not move unless you move, and you think the same way too. This is called a **Nash equilibrium**. In the prisoner's dilemma, `(Confess, Confess)` is a Nash equilibrium. In the coordination game, both `(Action, Action)` and `(Romance, Romance)` are Nash equilibria.

Symbolically, for a two-user game, suppose the two payoff functions are (U_1, U_2) and the two strategy spaces are $(\mathcal{A}, \mathcal{B})$ for the two players, respectively. We say $(a^* \in \mathcal{A}, b^* \in \mathcal{B})$ is a Nash equilibrium if

$$U_1(a^*, b^*) \geq U_1(a, b^*), \quad \text{for any } a \in \mathcal{A},$$

and

$$U_2(a^*, b^*) \geq U_2(a^*, b), \quad \text{for any } b \in \mathcal{B}.$$

A Nash equilibrium might not exist in a game. And when it exists, it need not be unique (like in the coordination game above), or socially optimal, or Pareto optimal. But if the players are allowed to throw a coin and decide *probabilistically* which strategy to play, i.e., a **mixed strategy**, it is guaranteed, by Nash's famous result, that a Nash equilibrium always exists.

We will expand and use our game-theoretic language in several future chapters. In our current case of power control as a game, the set of players is the set of transmitters. The strategy for each player is its transmit power level, and the strategy space is the set of transmit powers so that the target SIR is achieved. The cost function to minimize is the power level itself.

In this power control game, while the cost function is independent across the players, each player's strategy space A_i depends on the transmit powers of all the other players. This coupling across the players is introduced by the very nature of interference, and leads to strategic moves by the players.

Here is a simple fact, which is important to realize and easy to verify: iterating by the best response strategy of each player in the power control game is given precisely by (1.2). Given that all the other transmitters transmit at a certain power level, my best response strategy is to pick the power level that is my current power level times the ratio between my target and the current SIRs.

Now, consider player 1. If the transmit powers of all the other players become smaller, player 1's strategy space A_1 will be larger. (This is just another way to say there is negative externality.) There are more transmit powers to pick while still being able to maintain the target SIR, since other players' transmit powers are smaller and the denominator in the SIR is smaller. Given this sense of monotonicity, we suspect that maybe best responses converge. Indeed it can be shown that, for games with this property (and under some technicalities), the sequence of best response strategies converge.

This game-theoretic approach is one of the ways to prove the convergence of DPC. Another way is through the angle of global optimization and with the help of linear algebra, as we will present in the Advanced Material. But first, a small and detailed example.

1.3 Examples

A word about all the examples in this book. Most of them are completely numerical and presented in great detail, so as to alleviate any "symbol-phobia" a reader may have. This means that we have to strike a tradeoff between a realistic size for illustration and a small size to fit in a few pages. In some cases, these examples do not represent the actual scale of the problems, while scaling-up can actually be a core difficulty.

Suppose we have four (transmitter, receiver) pairs. Let the channel gains $\{G_{ij}\}$ be given in Table 1.3. As the table suggests, we can also represent these gains in a matrix. You can see that, in general, $G_{ij} \neq G_{ji}$ because the interference channels do not have to be symmetric.

Receiver of Link	Transmitter of Link 1	2	3	4
1	1	0.1	0.2	0.3
2	0.2	1	0.1	0.1
3	0.2	0.1	1	0.1
4	0.1	0.1	0.1	1

Table 1.3 Channel gains in an example of DPC. The entries are for illustrating the algorithm. They do not represent actual numerical values typically observed in real cellular networks.

Suppose that the initial power level is 1.0 mW on each link, and that the noise on each link is 0.1 mW. Then, the initial signal-to-interference ratios are given by

$$\begin{aligned}
\mathrm{SIR}_1[0] &= \tfrac{1\times1.0}{0.1\times1.0+0.2\times1.0+0.3\times1.0+0.1} = 1.43,\\
\mathrm{SIR}_2[0] &= \tfrac{1\times1.0}{0.2\times1.0+0.1\times1.0+0.1\times1.0+0.1} = 2.00,\\
\mathrm{SIR}_3[0] &= \tfrac{1\times1.0}{0.2\times1.0+0.1\times1.0+0.1\times1.0+0.1} = 2.00,\\
\mathrm{SIR}_4[0] &= \tfrac{1\times1.0}{0.1\times1.0+0.1\times1.0+0.1\times1.0+0.1} = 2.50,
\end{aligned}$$

where we use the formula

$$\mathrm{SIR}_i = \frac{G_{ii}p_i}{\sum_{j\neq i} G_{ij}p_j + n_i},$$

with p_i representing the power level of link i and n_i the noise on link i. These SIR values are shown on a linear scale rather than the log (dB) scale in this example.

We will use DPC to adjust the power levels. Suppose that the target SIRs are

$$\begin{aligned}
\gamma_1 &= 2.0,\\
\gamma_2 &= 2.5,\\
\gamma_3 &= 1.5,\\
\gamma_4 &= 2.0.
\end{aligned}$$

Then the new power levels are, in mW,

$$\begin{aligned}
p_1[1] &= \tfrac{\gamma_1}{\mathrm{SIR}_1[0]} p_1[0] = \tfrac{2.0}{1.43} \times 1.0 = 1.40,\\
p_2[1] &= \tfrac{\gamma_2}{\mathrm{SIR}_2[0]} p_2[0] = \tfrac{2.5}{2.00} \times 1.0 = 1.25,\\
p_3[1] &= \tfrac{\gamma_3}{\mathrm{SIR}_3[0]} p_3[0] = \tfrac{1.5}{2.00} \times 1.0 = 0.75,\\
p_4[1] &= \tfrac{\gamma_4}{\mathrm{SIR}_4[0]} p_4[0] = \tfrac{2.0}{2.5} \times 1.0 = 0.80.
\end{aligned}$$

Now each receiver calculates the new SIR and feeds it back to its transmitter:

$$\begin{aligned}
\mathrm{SIR}_1[1] &= \tfrac{1\times 1.40}{0.1\times 1.25+0.2\times 0.75+0.3\times 0.8+0.1} = 2.28,\\
\mathrm{SIR}_2[1] &= \tfrac{1\times 1.25}{0.2\times 1.40+0.1\times 0.75+0.1\times 0.8+0.1} = 2.34,\\
\mathrm{SIR}_3[1] &= \tfrac{1\times 0.75}{0.2\times 1.40+0.1\times 1.25+0.1\times 0.8+0.1} = 1.28,\\
\mathrm{SIR}_4[1] &= \tfrac{1\times 0.80}{0.1\times 1.40+0.1\times 1.25+0.1\times 0.75+0.1} = 1.82.
\end{aligned}$$

The new power levels in the next timeslot become, in mW,

$$\begin{aligned}
p_1[2] &= \tfrac{\gamma_1}{\mathrm{SIR}_1[1]} p_1[1] = \tfrac{2.0}{2.28} \times 1.40 = 1.23,\\
p_2[2] &= \tfrac{\gamma_2}{\mathrm{SIR}_2[1]} p_2[1] = \tfrac{2.5}{2.33} \times 1.25 = 1.34,\\
p_3[2] &= \tfrac{\gamma_3}{\mathrm{SIR}_3[1]} p_3[1] = \tfrac{1.5}{1.28} \times 0.75 = 0.88,\\
p_4[2] &= \tfrac{\gamma_4}{\mathrm{SIR}_4[1]} p_4[1] = \tfrac{2.0}{1.82} \times 0.80 = 0.88,
\end{aligned}$$

with the corresponding SIRs as follows:

$$\begin{aligned}
\mathrm{SIR}_1[2] &= \tfrac{1\times 1.23}{0.1\times 1.34+0.2\times 0.88+0.3\times 0.88+0.1} = 1.83,\\
\mathrm{SIR}_2[2] &= \tfrac{1\times 1.34}{0.2\times 1.23+0.1\times 0.88+0.1\times 0.88+0.1} = 2.56,\\
\mathrm{SIR}_3[2] &= \tfrac{1\times 0.88}{0.2\times 1.23+0.1\times 1.34+0.1\times 0.88+0.1} = 1.55,\\
\mathrm{SIR}_4[2] &= \tfrac{1\times 0.88}{0.1\times 1.23+0.1\times 1.34+0.1\times 0.88+0.1} = 1.98.
\end{aligned}$$

Calculating the new power levels again, we have, in mW,

$$\begin{aligned}
p_1[3] &= \tfrac{\gamma_1}{\mathrm{SIR}_1[2]} p_1[2] = \tfrac{2.0}{1.83} \times 1.23 = 1.35,\\
p_2[3] &= \tfrac{\gamma_2}{\mathrm{SIR}_2[2]} p_2[2] = \tfrac{2.5}{2.56} \times 1.34 = 1.30,\\
p_3[3] &= \tfrac{\gamma_3}{\mathrm{SIR}_3[2]} p_3[2] = \tfrac{1.5}{1.55} \times 0.88 = 0.85,\\
p_4[3] &= \tfrac{\gamma_4}{\mathrm{SIR}_4[2]} p_4[2] = \tfrac{2.0}{1.98} \times 0.88 = 0.89.
\end{aligned}$$

Then the new SIRs are

$$\begin{aligned}
\mathrm{SIR}_1[3] &= \tfrac{1\times 1.35}{0.1\times 1.30+0.2\times 0.85+0.3\times 0.89+0.1} = 2.02,\\
\mathrm{SIR}_2[3] &= \tfrac{1\times 1.30}{0.2\times 1.35+0.1\times 0.85+0.1\times 0.89+0.1} = 2.40,\\
\mathrm{SIR}_3[3] &= \tfrac{1\times 0.85}{0.2\times 1.35+0.1\times 1.30+0.1\times 0.89+0.1} = 1.45,\\
\mathrm{SIR}_4[3] &= \tfrac{1\times 0.89}{0.1\times 1.35+0.1\times 1.30+0.1\times 0.85+0.1} = 1.97,
\end{aligned}$$

and the new power levels, in mW, are

$$
\begin{aligned}
p_1[4] &= \tfrac{\gamma_1}{\mathrm{SIR}_1[3]} p_1[3] = \tfrac{2.0}{2.02} \times 1.35 = 1.33,\\
p_2[4] &= \tfrac{\gamma_2}{\mathrm{SIR}_2[3]} p_2[3] = \tfrac{2.5}{2.40} \times 1.30 = 1.36,\\
p_3[4] &= \tfrac{\gamma_3}{\mathrm{SIR}_3[3} p_3[3] = \tfrac{1.5}{1.45} \times 0.85 = 0.88,\\
p_4[4] &= \tfrac{\gamma_4}{\mathrm{SIR}_4[3]} p_4[3] = \tfrac{2.0}{1.97} \times 0.89 = 0.90.
\end{aligned}
$$

We see that the power levels are beginning to converge: p_1, p_2, p_3 and p_4 all change by less than 0.1 mW now. The new SIRs are

$$
\begin{aligned}
\mathrm{SIR}_1[4] &= \tfrac{1\times1.33}{0.1\times1.36+0.2\times0.88+0.3\times0.90+0.1} = 1.96,\\
\mathrm{SIR}_2[4] &= \tfrac{1\times1.36}{0.2\times1.33+0.1\times0.88+0.1\times0.90+0.1} = 2.49,\\
\mathrm{SIR}_3[4] &= \tfrac{1\times0.88}{0.2\times1.33+0.1\times1.36+0.1\times0.90+0.1} = 1.49,\\
\mathrm{SIR}_4[4] &= \tfrac{1\times0.90}{0.1\times1.33+0.1\times1.36+0.1\times0.88+0.1} = 1.97.
\end{aligned}
$$

Iterating one more time, the new power levels, in mW, are

$$
\begin{aligned}
p_1[5] &= \tfrac{\gamma_1}{\mathrm{SIR}_1[4]} p_1[4] = \tfrac{2.0}{1.96} \times 1.33 = 1.37,\\
p_2[5] &= \tfrac{\gamma_2}{\mathrm{SIR}_2[4]} p_2[4] = \tfrac{2.5}{2.49} \times 1.36 = 1.36,\\
p_3[5] &= \tfrac{\gamma_3}{\mathrm{SIR}_3[4]} p_3[4] = \tfrac{1.5}{1.49} \times 0.88 = 0.89,\\
p_4[5] &= \tfrac{\gamma_4}{\mathrm{SIR}_4[4]} p_4[4] = \tfrac{2.0}{1.97} \times 0.90 = 0.92,
\end{aligned}
$$

with corresponding SIRs:

$$
\begin{aligned}
\mathrm{SIR}_1[5] &= \tfrac{1\times1.37}{0.1\times1.36+0.2\times0.89+0.3\times0.92+0.1} = 1.98,\\
\mathrm{SIR}_2[5] &= \tfrac{1\times1.36}{0.2\times1.37+0.1\times0.89+0.1\times0.92+0.1} = 2.45,\\
\mathrm{SIR}_3[5] &= \tfrac{1\times0.89}{0.2\times1.37+0.1\times1.36+0.1\times0.92+0.1} = 1.48,\\
\mathrm{SIR}_4[5] &= \tfrac{1\times0.92}{0.1\times1.37+0.1\times1.36+0.1\times0.89+0.1} = 1.98.
\end{aligned}
$$

All the SIRs are now within 0.05 of the target. The power levels keep iterating, taking the SIRs closer to the target. Figure 1.5 shows the graph of power level versus the number of iterations. After about 20 iterations, the change is too small to be seen on the graph; the power levels at that time are

$$
\begin{aligned}
p_1 &= 1.46 \text{ mW},\\
p_2 &= 1.46 \text{ mW},\\
p_3 &= 0.95 \text{ mW},\\
p_4 &= 0.97 \text{ mW}.
\end{aligned}
$$

The resulting SIRs are shown in Figure 1.6. We get very close to the target SIRs, by visual inspection, after about 10 iterations.

While we are at this example, let us also walk through a compact, matrix representation of the target SIR constraints. This will be useful in the next section.

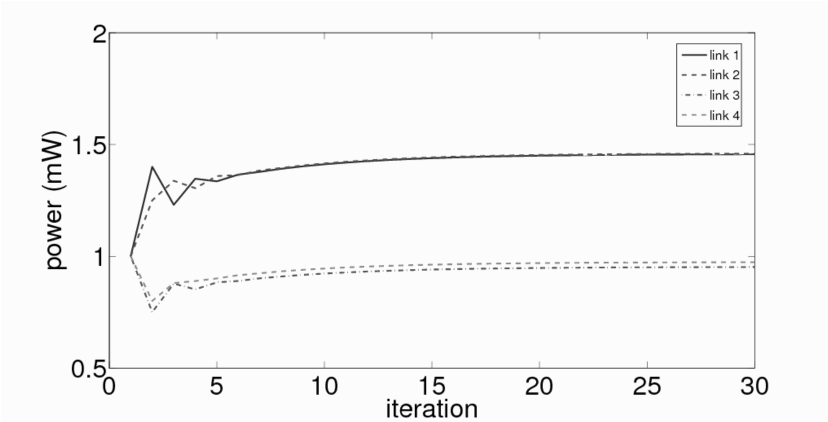

Figure 1.5 The convergence of power levels in an example of DPC.

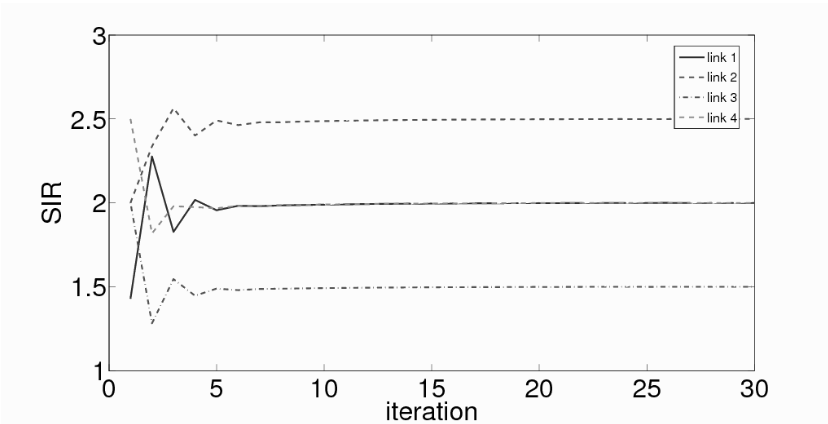

Figure 1.6 The convergence of SIRs in an example of DPC.

If the target SIR γ_i is achieved or exceeded by $\{p_i\}$, we have

$$\frac{G_{ii}p_i}{\sum_{j \neq i} G_{ij}p_j + n_i} \geq \gamma_i,$$

for $i = 1, 2, 3, 4$. On multiplying both sides by $\sum_{j \neq i} G_{ij}p_j + n_i$ and dividing by G_{ii}, we have

$$p_i \geq \frac{\gamma_i}{G_{ii}} \left(\sum_{j \neq i} G_{ij}p_j + n_i \right),$$

which can be written as

$$p_i \geq \gamma_i \sum_{j \neq i} \frac{G_{ij}}{G_{ii}} p_j + \frac{\gamma_i}{G_{ii}} n_i. \tag{1.4}$$

Now we define the variable vector

$$\mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix},$$

and a constant vector

$$\mathbf{v} = \begin{bmatrix} \frac{\gamma_1 n_1}{G_{11}} \\ \frac{\gamma_2 n_2}{G_{22}} \\ \frac{\gamma_3 n_3}{G_{33}} \\ \frac{\gamma_4 n_4}{G_{44}} \end{bmatrix} = \begin{bmatrix} \frac{2.0\times 0.1}{1.0} \\ \frac{2.5\times 0.1}{1.0} \\ \frac{1.5\times 0.1}{1.0} \\ \frac{2.0\times 0.1}{1.0} \end{bmatrix} = \begin{bmatrix} 0.20 \\ 0.25 \\ 0.15 \\ 0.20 \end{bmatrix}.$$

Define also a 4×4 diagonal matrix $\mathbf{D}$ with γ_i on the diagonal, and another 4×4 matrix $\mathbf{F}$ where $F_{ij} = G_{ij}/G_{ii}$ for $i \neq j$, and the diagonal entries of F are zero. Plugging in the numbers, we have

$$\mathbf{D} = \begin{bmatrix} 2.0 & 0 & 0 & 0 \\ 0 & 2.5 & 0 & 0 \\ 0 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 2.0 \end{bmatrix},$$

$$\mathbf{F} = \begin{bmatrix} 0 & 0.1 & 0.2 & 0.3 \\ 0.2 & 0 & 0.1 & 0.1 \\ 0.2 & 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0 \end{bmatrix}.$$

We can now rewrite (1.4) as

$$\mathbf{p} \geq \mathbf{DFp} + \mathbf{v} = \begin{bmatrix} 0 & 0.20 & 0.40 & 0.60 \\ 0.50 & 0 & 0.25 & 0.25 \\ 0.30 & 0.15 & 0 & 0.15 \\ 0.20 & 0.20 & 0.20 & 0 \end{bmatrix} \mathbf{p} + \begin{bmatrix} 0.20 \\ 0.25 \\ 0.15 \\ 0.20 \end{bmatrix},$$

where $\geq$ between two vectors (of equal length) simply represents component-wise inequality between the corresponding entries of the two vectors.

We can check that the power levels in the last iteration shown above satisfy this inequality tightly:

$$\begin{bmatrix} 1.46 \\ 1.46 \\ 0.95 \\ 0.97 \end{bmatrix} \geq \begin{bmatrix} 0 & 0.20 & 0.40 & 0.60 \\ 0.50 & 0 & 0.25 & 0.25 \\ 0.30 & 0.15 & 0 & 0.15 \\ 0.20 & 0.20 & 0.20 & 0 \end{bmatrix} \begin{bmatrix} 1.46 \\ 1.46 \\ 0.95 \\ 0.97 \end{bmatrix} + \begin{bmatrix} 0.20 \\ 0.25 \\ 0.15 \\ 0.20 \end{bmatrix} = \begin{bmatrix} 1.46 \\ 1.46 \\ 0.95 \\ 0.97 \end{bmatrix}.$$

We will use this matrix representation as we go from problem representation (1.3) to problem representation (1.5) in the next section. We will also see in later chapters many different matrices that summarize the *topology* of a network, and operations involving these matrices that model *functionalities* running on the network.

1.4 Advanced Material

1.4.1 Iterative power method

We can generalize the vector notation in the last example. Let $\mathbf{1}$ represent a vector of 1s, so the objective function is simply $\mathbf{1}^T\mathbf{p} = \sum_i p_i$. Let $\mathbf{I}$ be the identity matrix, $\mathbf{D}$ a diagonal matrix with the diagonal entries being the target SIR values $\{\gamma_i\}$, and $\mathbf{F}$ be a matrix capturing the given channel conditions: $F_{ij} = G_{ij}/G_{ii}$ if $i \neq j$, and $F_{ii} = 0$. The constant vector $\mathbf{v}$ captures the normalized noise levels, with $v_i = \gamma_i n_i / G_{ii}$. We will soon see why this shorthand notation is useful.

Equipped with the notation above, we can represent the target SIR constraints in problem (1.3) as

$$\mathbf{p} \geq \mathbf{DFp} + \mathbf{v},$$

and further group all the terms involving the variables $\mathbf{p}$. The linear programming problem we have now becomes

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T\mathbf{p} \\ \text{subject to} & (\mathbf{I} - \mathbf{DF})\mathbf{p} \geq \mathbf{v} \\ \text{variables} & \mathbf{p}. \end{array} \tag{1.5}$$

You should verify that problem (1.3) and problem (1.5) are indeed equivalent, by using the definitions of the SIR, of matrices $(\mathbf{D}, \mathbf{F})$, and of vector $\mathbf{v}$.

Linear programming problems are conceptually and computationally easy to solve in general. Our special case here has even more structure. This $\mathbf{DF}$ matrix is a **non-negative matrix**, since all entries of the matrix are non-negative numbers. Non-negative matrices are a powerful modeling tool in linear algebra, with applications from economics to ecology. They have been well-studied in matrix analysis through the **Perron–Frobenius theory**.

If the largest eigenvalue of the $\mathbf{DF}$ matrix, denoted as $\rho(\mathbf{DF})$, is less than 1, then the following three statements are true.

- (a) We can guarantee that the set of target SIRs can indeed be achieved simultaneously, which makes sense since $\rho(\mathbf{DF}) < 1$ means that the $\{\gamma_i\}$ in $\mathbf{D}$ are not "too big," relative to the given channel conditions captured in $\mathbf{F}$.
- (b) We can invert the matrix defining the linear constraints in our optimization problem (1.5): solve the problem by computing $(\mathbf{I}-\mathbf{DF})^{-1}\mathbf{v}$. But, of course, there is no easy way to directly run this matrix inversion distributively across the MSs.
- (c) The inversion of the matrix can be expressed as a sum of terms, each term a multiplication of matrix $\mathbf{DF}$ by itself. More precisely: $(\mathbf{I} - \mathbf{DF})^{-1} = \sum_{k=0}^{\infty}(\mathbf{DF})^k$. This is an infinite sum, so we say that the partial sum of K

terms, $\sum_{k=0}^{K}(\mathbf{DF})^k$, will converge as K becomes very large. Furthermore, the tail term in this sum, $(\mathbf{DF})^k$, approaches 0 as k becomes large:

$$\lim_{k\to\infty}(\mathbf{DF})^k = 0.$$

The key insight is that we want to invert a matrix $(\mathbf{I}-\mathbf{DF})$, because that will lead us to a power-minimal solution to achieving all the target SIRs:

$$\mathbf{p}^* = (\mathbf{I}-\mathbf{DF})^{-1}\mathbf{v}$$

is a solution of problem (1.3), i.e., for any solution $\hat{\mathbf{p}}$ satisfying the constraints in problem (1.3), $\mathbf{p}^*$ is better:

$$\mathbf{p}^* \leq \hat{\mathbf{p}}.$$

Now, the matrix-inversion step is not readily implementable in a distributed fashion, as we need in cellular network power control. Fortunately, it can be achieved by applying the following update.

(1) First, $\mathbf{p}^* = (\mathbf{I}-\mathbf{DF})^{-1}\mathbf{v}$ can be represented as a power series, as stated in Statement (c) above:

$$\mathbf{p}^* = \sum_{k=0}^{\infty}(\mathbf{DF})^k\mathbf{v}. \tag{1.6}$$

(2) Then, you can readily check that the following iteration over time gives exactly the above power series (1.6), as time t goes on:

$$\mathbf{p}[t+1] = \mathbf{DF}\mathbf{p}[t] + \mathbf{v}. \tag{1.7}$$

One way to check this is to substitute the above recursive formula of $\mathbf{p}$ (1.7) all the way to $\mathbf{p}[0]$, the initialization of the iteration. Then, at any time t, we have

$$\mathbf{p}[t] = (\mathbf{DF})^t\mathbf{p}[0] + \sum_{k=0}^{t-1}(\mathbf{DF})^k\mathbf{v},$$

which converges, as $t \to \infty$, to

$$\lim_{t\to\infty}\mathbf{p}[t] = 0 + \sum_{k=0}^{\infty}(\mathbf{DF})^k\mathbf{v} = \mathbf{p}^*,$$

since $(\mathbf{DF})^t\mathbf{p}[0]$ approaches 0 as $t \to \infty$. So, we know (1.7) is right. This also shows that it does not matter what the initialization vector $\mathbf{p}[0]$ is. Its effect will be washed away as time goes on.

(3) Finally, rewrite the vector form of update equation (1.7) in scalar form for each transmitter i, and you will see that it is exactly the DPC algorithm (1.2):

$$p_i[t+1] = \frac{\gamma_i}{\mathrm{SIR}_i[t]}p_i[t], \quad \text{for each } i.$$

We just completed a development and convergence analysis of the DPC as the solution of a global optimization problem through the language of linear algebra. In general, for any square matrix $\mathbf{A}$, the following statements are equivalent.

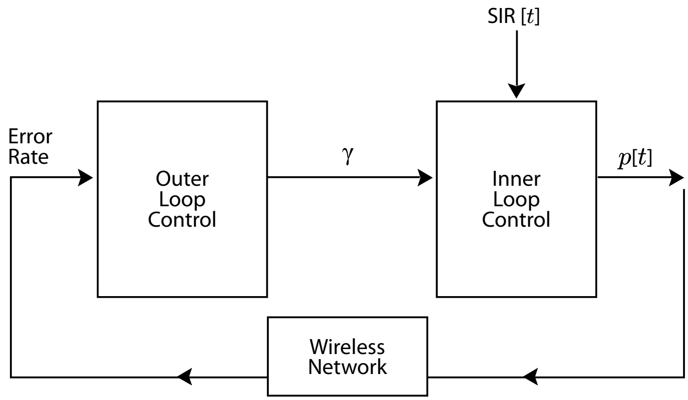

Figure 1.7 Inner and outer loops of power control in cellular networks. The inner loop takes in a fixed target SIR, compares it with the current SIR, and updates the transmit power. The outer loop adjusts the target SIR on the basis of the performance measured over a longer timescale. We have focused on the inner-loop power control in this chapter.

1. The largest eigenvalue is less than 1.
2. The limit of this matrix multiplying itself is 0: $\lim_{k\to\infty} \mathbf{A}^k = 0$.
3. The infinite sum of powers $\sum_{k=0}^{\infty} \mathbf{A}^k$ exists and equals $(\mathbf{I} - \mathbf{A})^{-1}$.

What we saw in DPC is an iterative **power method** (the word "power" here has nothing to do with transmit powers, but instead refers to raising a matrix to some power, i.e., multiplying a matrix by itself many times). It is a common method used to develop iterative algorithms arising from linear systems models. We formulate the problem as a linear program, then we define the solution to the problem as the solution to a linear equation, implement the matrix inversion through a sequence of matrix powers, turn each of those steps into an easy computation at each timeslot, and, finally, achieve the matrix inversion through an iteration in time. This will be used again when we talk about Google's PageRank algorithm in Chapter 3.

1.4.2 Outer loop power control

As shown in the block diagram in Figure 1.7, in cellular networks there are two timescales of power control. What we have been discussing so far is the **inner-loop power control**. Nested outside of that is the **outer-loop power control**, where the target SIRs $\{\gamma_i\}$ are determined.

One standard way to determine target SIRs is to measure the received signal quality, in terms of decoding error probabilities, at the receiver. If the error rate is too high, the target SIR needs to be increased. And if the error rate is lower than necessary, the target SIR can be reduced.

Alternatively, we can also consider optimizing target SIRs as optimization *variables*. This is particularly useful for 3G and 4G networks where data traffic dominates voice traffic in cellular networks. A higher SIR can provide a higher rate at the same signal quality. But every user wants to achieve a higher SIR. So we need to model this either as a network-wide optimization, maximizing the sum of all users' payoff functions, or as a game, with each user maximizing its own payoff function of SIR.

Summary

Box 1 Distributed power control

Different users' signals interfere with each other in the air, leading to a feasible SIR region with a Pareto-optimal boundary. Interference coordination in CDMA networks can be achieved through distributed power control with implicit feedback. It solves an optimization problem for the network in the form of linear programming, and can also be modeled as a non-cooperative game.

Further Reading

This chapter introduces several foundational methodologies: optimization, games, and algorithms, as well as the basics of cellular wireless networks. As a result, there are many interesting texts to read.

1. An early paper quantifying the benefits of CDMA was written by a Qualcomm team, including Andrew Viterbi:
I. M. Jacobs, R. Padovani, A. J. Viterbi, L. A. Weaver, C. E. Wheatley, "On the capacity of a cellular CDMA system," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 2, pp. 303–312, May 1991.

2. The DPC algorithm in the core of this chapter appeared in the following seminal paper:
G. J. Foschini and Z. Miljanic, "A simple distributed autonomous power control algorithm and its convergence," *IEEE Transactions on Vehicular Technology*, vol. 42, no. 3, pp. 641–646, November 1993.

3. Much more discussion on power control algorithms in cellular networks can be found in the following monograph:
M. Chiang, P. Hande, T. Lan, and C. W. Tan, "Power control for cellular wireless networks," *Foundation and Trends in Networking*, vol. 2, no. 4, pp. 381-533, July 2008.

4. A standard reference on linear algebra is the following mathematics textbook, which includes a chapter on non-negative matrices and Perron–Frobenius theory:

R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1990.

5. There are many textbooks on game theory, in all types of styles. A comprehensive introduction is

R. B. Myerson, *Game Theory: Analysis of Conflict*, Harvard University Press, 1997.

Problems

1.1 *Distributed power control* $\star$

(a) Consider three pairs of transmitters and receivers in a cell, with the following channel gain matrix $\mathbf{G}$ and noise of 0.1 mW for all the receivers. The target SIRs are also shown below.

$$\mathbf{G} = \begin{bmatrix} 1 & 0.1 & 0.3 \\ 0.2 & 1 & 0.3 \\ 0.2 & 0.2 & 1 \end{bmatrix}, \qquad \gamma = \begin{bmatrix} 1 \\ 1.5 \\ 1 \end{bmatrix}.$$

With an initialization of all transmit powers at 1 mW, run DPC for ten iterations and plot the evolution of transmit powers and received SIRs. You can use any programming language, or even write the steps out by hand.

(b) Now suppose the power levels for logical links 1, 2, and 3 have converged to the equilibrium in (a). A new pair of transmitter and receiver, labeled as logical link 4, shows up in the same cell, with an initial transmit power of 1 mW and demands a target SIR of 1. The new channel gain matrix is shown below.

$$\mathbf{G} = \begin{bmatrix} 1 & 0.1 & 0.3 & 0.1 \\ 0.2 & 1 & 0.3 & 0.1 \\ 0.2 & 0.2 & 1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 1 \end{bmatrix}.$$

Similarly to what you did in (a), show what happens in the next ten timeslots. What happens at the new equilibrium?

1.2 *Power control infeasibility* $\star\star$

Consider a three-link cell with the link gains G_{ij} shown below. The receivers request $\gamma_1 = 1, \gamma_2 = 2$, and $\gamma_3 = 1$. The noise $n_i = 0.1$ for all i.

$$\mathbf{G} = \begin{bmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{bmatrix}.$$

Prove this set of target SIRs is infeasible.

1.3 *A zero-sum game* $\star$

In the following two-user game, the payoffs of users Alice and Bob are exactly negative of each other in all the combinations of strategies (a,a), (a,b), (b,a), (b,b). This models an extreme case of competition, and is called a **zero-sum game**. Is there any pure strategy equilibrium? How many are there?

	a	b
a	(2, −2)	(3, −3)
b	(3, −3)	(4, −4)

1.4 *Mechanism design* $\star\star$

Consider the game below. There are two players, Alice and Bob, each with two strategies, and the payoffs are shown below. Consider only pure strategy equilibria.

	a	b
a	(0, 2)	(2, 0)
b	(6, 0)	(3, 2)

(a) Is there a Nash equilibrium, and, if so, what is it?

(b) We want to make this game a "better" one. What entries in the table would you change to make the resulting Nash equilibrium unique and socially optimal?

This is an example of **mechanism design**: change the game so as to induce movement of players to a desirable equilibrium. We will see a lot more of mechanism design in future chapters.

1.5 *Repeating prisoner's dilemma* $\star\star\star$

(a) Suppose the two prisoners know that they will somehow be caught in the same situation five more times in future years. What will each prisoner's strategy be in choosing between confession and no confession?

(b) Suppose the two prisoners have infinite lifetimes, and there is always a 90% chance that they will be caught in the same situation after each round of this game. What will each prisoner's strategy be now?