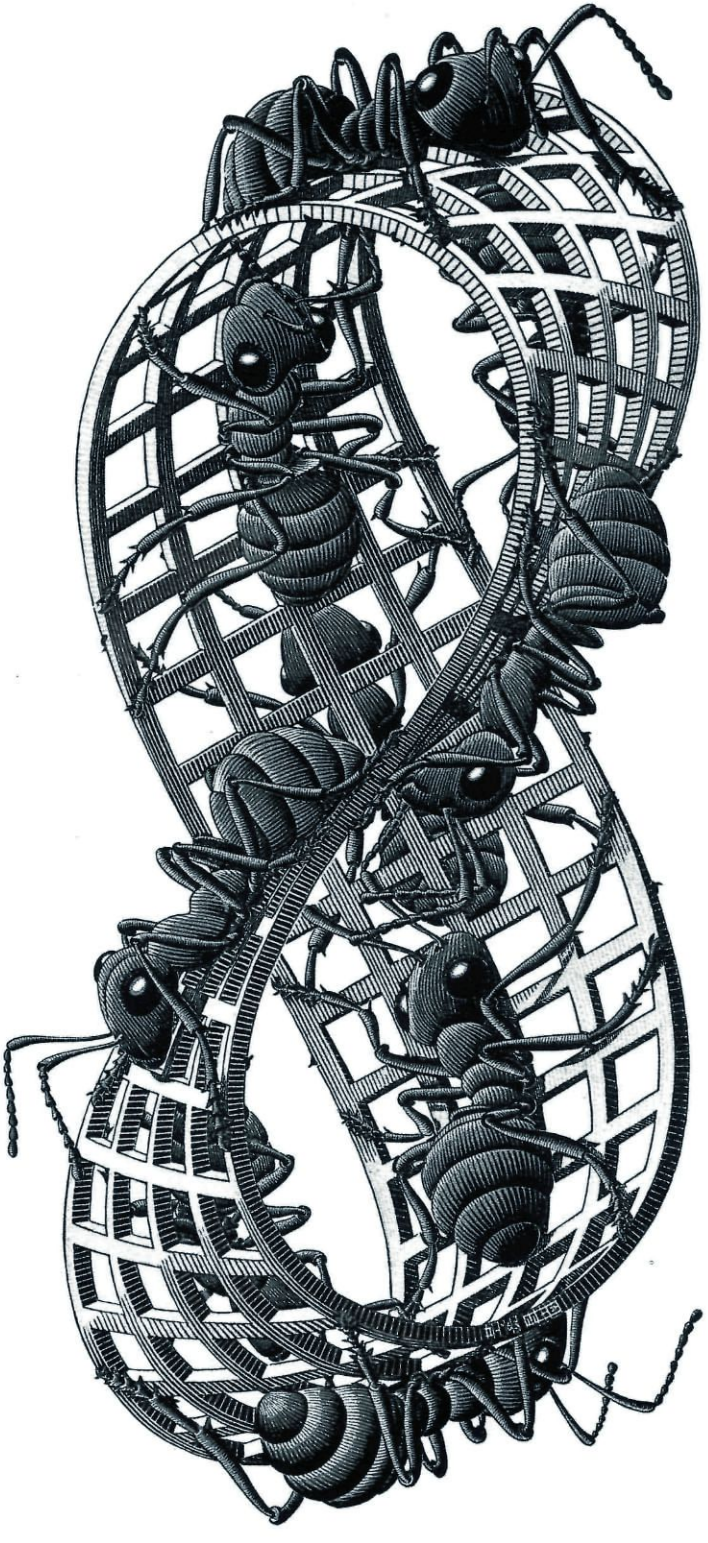
Preludio . . .

Aquiles y la Tortuga han ido a casa de su amigo el Cangrejo a fin de conocer a uno de los amigos de éste, el Oso Hormiguero. Ya han sido hechas las presentaciones, y los cuatro se encuentran sentados, tomando el té.

Tortuga: Le hemos traído una cosita, señor Cangrejo.
Cangrejo: Qué amables, pero no debieran haberse molestado.
Tortuga: No es más que una muestra de nuestra estima. Aquiles, ¿querría entregársela al señor Cangrejo?
Aquiles: Por supuesto. Con nuestros mejores deseos. Que la disfrute usted, señor Cangrejo.

(Aquiles entrega al Cangrejo un elegante envoltorio, de forma cuadrada y muy delgado. El Cangrejo comienza a desenvolverlo.)

Oso Hormiguero: Qué podrá ser . . .
Cangrejo: Pronto lo sabremos. (*Termina de desenvolverlo, y extrae el regalo.*) ¡Dos discos! ¡Qué emocionante! Pero no tienen rótulo. Ah, eh . . . ¿es otro de sus presentes "especiales", señora T?
Tortuga: Si se refiere usted a un destructor de fonógrafos, no se trata de ello en esta oportunidad, pero sí de un artículo de ese mismo ramo, el único de su clase en el mundo entero. En realidad, es algo que nunca fue escuchado hasta ahora . . . salvo, claro está, cuando Bach lo ejecutó.
Cangrejo: ¿Cuando Bach lo ejecutó? ¿Qué quiere decir usted, exactamente?
Aquiles: Oh, se sentirá usted fabulosamente conmovido, señor Cangrejo, cuando la señora Tortuga le diga qué son verdaderamente estos discos.
Tortuga: Pero, no se detenga, Aquiles, y dígalo usted.
Aquiles: Oh, ¿sí? Ay, caramba. Será mejor que consulte mis apuntes, entonces. (*Extrae una pequeña ficha, y se aclara la voz.*) Ejem. ¿Les interesaría que hablásemos del reciente y notable descubrimiento matemático al que estos discos deben su existencia?
Cangrejo: ¿Estos discos derivan de alguna cuestión matemática? ¡Qué curioso! Bien, ya ha conseguido usted despertar mi interés, así que continúe, por favor.
Aquiles: Perfectamente, pues. (*Se detiene por un momento, bebe un sorbo de té, y luego prosigue.*) ¿Han oído nombrar el polémico "Teorema Final" de Fermat?
Oso Hormiguero: No estoy seguro . . . Me suena curiosamente familiar, pero no alcanzo a ubicarlo.



Aquiles: Se trata de una idea sumamente simple. Pierre de Fermat, abogado de profesión y matemático por distracción, estaba leyendo el clásico texto de Diofanto, *Arithmetica*, en una de cuyas páginas se topó con la siguiente ecuación:

$$a^2 + b^2 = c^2$$

Comprendió de inmediato que esta ecuación tiene infinitas soluciones a, b, c, y entonces escribió, al margen, este notable comentario:

La ecuación

$$a^n + b^n = c^n$$

tiene soluciones, a través de los enteros positivos a, b, c y n, únicamente si n = 2 (y entonces habrá infinitos tripletes a, b, c, que satisfagan la ecuación); pero no hay soluciones si n>2. He descubierto una demostración verdaderamente prodigiosa de esta afirmación, pero, desdichadamente, este margen es demasiado pequeño para contenerla.

Desde ese día, unos trescientos años atrás, los matemáticos se han estado esforzando por conseguir una cosa o la otra: o bien probar lo afirmado por Fermat, y en consecuencia reivindicar su reputación, la cual, aunque muy alta, ha sido un tanto empañada por los escépticos para quienes Fermat, en realidad, jamás encontró la demostración que dijo haber encontrado; o bien refutar su afirmación, hallando el contraejemplo adecuado: un conjunto de cuatro enteros a, b, c y n, donde n>2, que satisfaga la ecuación. Hasta hace muy poco, todos los ensayos en ambas direcciones habían fracasado. Existe la seguridad de que el Teorema ya ha sido probado con relación a muchos valores específicos de n: en particular, todos los valores n hasta 125000.

Oso Hormiguero: ¿No habría que llamarlo "Conjetura", en vez de "Teorema", puesto que nunca ha sido adecuadamente demostrado?

Aquiles: Estrictamente hablando, tiene usted razón; es por tradición que se le dice "Teorema".

Cangrejo: ¿No hay nadie que se las haya ingeniado para resolver este célebre problema?

Aquiles: ¡Por cierto que sí! La señora Tortuga lo ha conseguido y, como es usual en ella, a través de un toque mágico. No solamente ha descubierto una DEMOSTRACION del Teorema Final de Fermat (dando justificación, así, a tal denominación, y reivindicando al mismo tiempo el prestigio de Fermat), sino también un CONTRAEJEMPLO, que muestra la excelente intuición de los escépticos . . .

Cangrejo: ¡Caramba! Sí que es un hallazgo revolucionario.

Figura 54. Banda de Moebius II, *de M. C. Escher (grabado en madera, 1963).*

Figura 55. Pierre de Fermat.

Oso Hormiguero: ¡Por favor, basta de suspenso! ¿Cuáles son esos mágicos enteros que satisfacen la ecuación de Fermat? Siento especial curiosidad con respecto al valor de n.

Aquiles: ¡Oh, qué horror! ¡Estoy apenadísimo! ¿Pueden creerlo? Me dejé los valores en casa, pues era un montón verdaderamente colosal de papel. Por desdicha, era algo demasiado enorme para traerlo conmigo. Desearía tenerlo acá para que pudiesen verlo. Pero, si de algo sirve, me acuerdo de una cosa: el valor de n es el único entero positivo que no aparece nunca en la fracción continua π.

Cangrejo: Oh, cuán lamentale es que no tengamos eso ahora. Sin embargo, no hay razón para poner en duda lo que usted nos ha dicho.

Oso Hormiguero: De cualquier modo, ¿para qué queremos ver a n enunciado decimalmente? Aquiles nos ha indicado cómo encontrarlo. ¡Bien, señora Tortuga, le ruego que acepte mis sinceras felicitaciones por su trascendental descubrimiento!

Tortuga: Muchas gracias. Pero lo que me parece más importante que el resultado mismo es la utilización práctica a la cual conduce directamente mi resultado.

Cangrejo: Me estoy muriendo por escucharla hablar de eso, pues siempre pensé que la teoría de los números era la Reina de la Matemática —la rama más pura de la matemática—, la única rama de la matemática que NO tiene aplicaciones . . .

Tortuga: No es usted la única persona que cree eso. No obstante, es del todo imposible hacer una afirmación genérica acerca de cuándo o cómo una rama determinada de la matemática pura —o, inclusive, determinado Teorema en particular— obtendrá repercusiones importantes fuera de la matemática. Es algo enteramente impredictible, y el pre-

sente caso es un ejemplo perfecto de tal fenómeno.

Aquiles: ¡El descubrimiento doble cañón de la señora Tortuga ha abierto un camino en el campo de la caza acústica!

Oso Hormiguero: ¿Qué es la caza acústica?

Aquiles: El nombre ya lo dice: es la caza o recuperación de información acústica en fuentes extremadamente complejas. Un caso ejemplificador de caza acústica es la reconstrucción del sonido producido por un peñasco que cae a plomo en un lago, a partir de las ondas que se esparcen sobre la superficie del agua.

Cangrejo: ¡Caray, eso parece casi imposible!

Aquiles: No crea. En realidad, es algo enteramente similar a lo que hace nuestro cerebro cuando, a partir de las vibraciones transmitidas por el tímpano a las fibras de la cóclea, reconstruye el sonido producido en las cuerdas vocales de otra persona.

Cangrejo: Ya veo. Pero lo que todavía no veo es cómo entra la teoría de los números en este cuadro, ni qué tienen que ver mis nuevos discos con todo esto.

Aquiles: Bueno, en la matemática de la recuperación acústica surgen muchos problemas que están relacionados con las diversas soluciones de ciertas ecuaciones diofantinas. Por su parte, la señora Tortuga ha dedicado años a tratar de reconstruir los sonidos producidos por Bach al ejecutar su clave, hace más de dos siglos, a partir del cálculo, hoy, de los movimientos de todas las moléculas en la atmósfera.

Oso Hormiguero: ¡Pero eso es imposible! ¡Esos sonidos son irrecuperables, se han ido para siempre!

Aquiles: Así piensa el ingenuo . . . Sin embargo, la señora T ha estudiado durante muchos años este problema, y llegó a la conclusión de que todo depende de las diversas soluciones que tenga la ecuación

$$a^n + b^n = c^n$$

en enteros positivos, siendo $n>2$.

Tortuga: Precisamente, yo podría explicar cómo surge esta correspondencia, pero estoy segura de que se van a aburrir.

Aquiles: Resulta que la teoría de la recuperación acústica predice que los sonidos de Bach pueden ser recuperados a partir del movimiento de todas las moléculas en la atmósfera, si es posible decir que, O existe por lo menos una solución a la ecuación . . .

Cangrejo: ¡Asombroso!

Oso Hormiguero: ¡Fantástico!

Tortuga: ¡Quién lo hubiera pensado!

Aquiles: Yo estaba diciendo, "¡si es posible decir que, O existe tal solución O existe una demostración de que NO hay solución!" En consecuencia, la señora Tortuga empezó a trabajar, de manera muy cuidadosa, sobre los dos polos del problema a la vez. Cuando se produjo el descubrimiento

del contraejemplo, éste sirvió de ingrediente clave para hallar la demostración, de modo que lo primero condujo directamente a lo segundo.

Cangrejo: ¿Cómo pudo ser eso?

Tortuga: Bueno, fíjese, yo había mostrado que la organización estructural de cualquier demostración del Teorema Final de Fermat —si es que existía alguna— podía ser descripta mediante una fórmula precisa, la cual, tal como efectivamente sucedió, dependería de los valores de la solución que se hallase para determinada ecuación. Cuando encontré esta segunda ecuación, la misma resultó ser, para mi sorpresa, la ecuación de Fermat: una graciosa relación accidental entre forma y contenido. Entonces, cuando descubrí el contraejemplo, todo lo que necesité hacer fue utilizar aquellos números para construir mi demostración de que no hay soluciones a la ecuación. Notablemente simple, si se lo piensa. No puedo explicarme por qué nadie obtuvo antes este resultado.

Aquiles: Como consecuencia de este logro matemático incalculablemente valioso, la señora Tortuga estuvo en condiciones de concretar la recuperación acústica con la que tanto tiempo había soñado. Y este regalo recibido hoy por el señor Cangrejo representa una materialización tangible de todo el trabajo abstracto mencionado.

Cangrejo: ¡No me diga que se trata de grabaciones de Bach ejecutando sus propias obras para clave!

Aquiles: Lo lamento, pero eso mismo es lo que debo decirle, pues se trata exactamente de lo que usted supuso... Son dos grabaciones de Juan Sebastián Bach ejecutando su composición *El Clave Bien Temperado*. Cada disco contiene uno de los dos volúmenes de la obra, es decir, 24 preludios y fugas, en tonalidad mayor en uno de los discos, y en tonalidad menor en el otro.

Cangrejo: ¡Bueno, no puedo pensar sino en escuchar de inmediato una de estas inapreciables grabaciones! ¿Cómo podré agradecerles?

Tortuga: Su agradecimiento ya está plenamente mostrado, con este delicioso té que nos sirvió.

(El Cangrejo quita el envoltorio a uno de los discos, y dispone lo necesario para escucharlo. Instantes después, el sonido de una ejecución increíblemente magistral llena el cuarto, con la más alta fidelidad imaginable. Hasta se puede oír —¿o no será más que ilusión?— el suave sonido de la voz de Bach, tarareando para sí mismo mientras toca . . .)

Cangrejo: ¿Alguno de ustedes querría seguir la música con la partitura? Sucede que tengo una edición única de *El Clave Bien Temperado*, especialmente ilustrada por un maestro mío a quien también le sucedía que era un finísimo calígrafo.

Tortuga: Me encantaría.

(El Cangrejo va hasta su elegante biblioteca de madera, protegida con cristales; abre las puertas, y retira dos grandes tomos.)

Cangrejo: Aquí tiene, señora Tortuga. En verdad, nunca me he sentido llevado a conocer todas las bellas ilustraciones de esta edición. Quizá su regalo me impulse ahora a hacerlo.

Tortuga: Así lo espero.

Oso Hormiguero: ¿Se han dado cuenta de cómo, en estas composiciones, los preludios siempre disponen el ánimo adecuadamente para la fuga que sigue?

Cangrejo: Sí. Aunque es muy difícil expresarla con palabras, siempre hay cierta relación sutil entre ambos. Aun cuando el preludio y la fuga no tengan el mismo tema melódico, en todos los casos existe una inefable cualidad abstracta que subyace a ambos, ligándolos así vigorosamente.

Tortuga: Y los escasos instantes de ansiedad silenciosa suspendidos entre preludio y fuga acumulan un gran dramatismo: esos instantes en que el tema de la fuga está por ser anunciado, en notas separadas, que luego se entremezclarán, a través de niveles más y más complejos de misteriosa y exquisita armonía.

Aquiles: Sé muy bien a qué se refiere usted. Hay muchos preludios y fugas que todavía no conozco, y entonces ese fugaz silencio me llena de anhelo pues durante su transcurso trato de conjeturar qué nos dará el viejo Bach. Por ejemplo, siempre me pregunto si el tempo de la fuga será allegro o adagio; si estará en 6/8 o en 4/4; si habrá tres voces, o cinco . . . o cuatro. Y después, comienza la primera voz . . . Son momentos deliciosos.

Cangrejo: Ah, sí, qué bien recuerdo los lejanos días de mi juventud, cuando me estremecía con cada preludio y fuga, los cuales me invadían con la excitación de su novedad y de su belleza, y de las muchas y grandes sorpresas que encerraban.

Aquiles: ¿Y ahora? ¿Es que ese estremecimiento ya no se produce?

Cangrejo: Ha sido suplantado por la familiaridad, sin desaparecer por ello. Y en esta familiaridad hay una hondura compensatoria. Por ejemplo, siempre me encuentro con sorpresas, con elementos que antes no había advertido.

Aquiles: ¿Apariciones del tema que había pasado por alto, quizá?

Cangrejo: Es posible . . . especialmente cuando aquél es invertido y ocultado en el interior de varias otras voces, o cuando parece irrumpir de súbito desde las profundidades, o desde la nada. Pero hay también modulaciones pasmosas que es maravilloso escuchar una y otra vez, y de las que uno se pregunta a través de qué prodigio las concibió el viejo Bach.

Aquiles: Me complace mucho saber que se pueden esperar más cosas, luego de experimentar mi primer rapto de fascinación ante *El Clave Bien Temperado* . . . sin embargo, también me entristece pensar que este estadio no pueda prolongarse por siempre jamás.

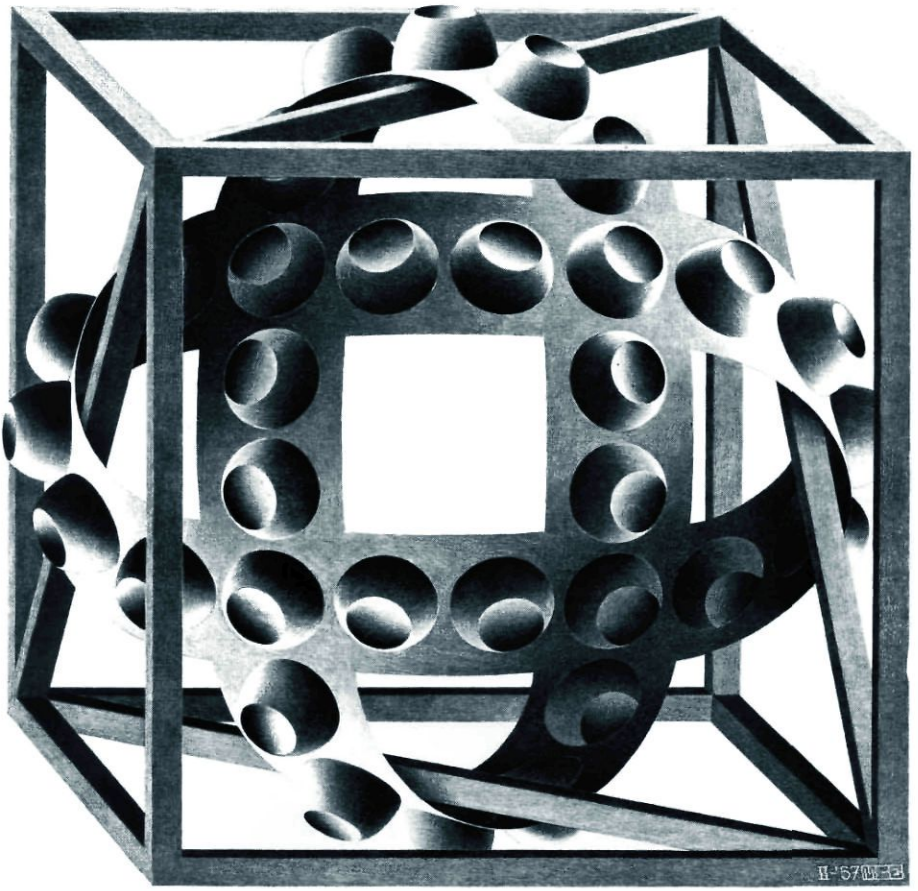


Figura 56. Cubo con Bandas Mágicas, *de M. C. Escher (litografía, 1957).*

Cangrejo: Oh, no tiene usted por qué temer que muera totalmente su arrobamiento. Una de las cualidades más preciosas de aquel estremecimiento juvenil es que siempre puede resucitar, precisamente en el momento en que uno había llegado a creer que había muerto definitivamente. Basta con que actúe, desde fuera, el disparador adecuado.

Aquiles: ¿De veras? ¿Qué puede actuar así?

Cangrejo: El escuchar la música, por decir así, a través de los oídos de alguien para quien ello constituye una experiencia totalmente nueva: de alguien como usted, Aquiles. De alguna manera, el estremecimiento consigue transmitirse, y yo puedo sentir que mi vieja conmoción resucita.

Aquiles: Esto es curioso. Esa reacción ha permanecido en latencia en algún sitio dentro suyo, pero usted no puede, por sí mismo, sacarla de su subconsciente.

Cangrejo: Exactamente. La posibilidad de revivir el estremecimiento está "codificada", a través de alguna forma desconocida, en el interior

de la estructura de mi cerebro, pero yo no estoy facultado para convocarla a voluntad. Tengo que esperar la ocurrencia de una circunstancia casual que la dispare.

Aquiles: Hay una pregunta que me gustaría hacer, con respecto a las fugas, pero me avergüenza un poco tener que formularla; claro que, como soy apenas un principiante en la audición de fugas, pensé que tal vez alguno de ustedes, ya maduro en esto, podría ayudarme con sus enseñanzas . . .

Tortuga: Ciertamente, me agradaría ofrecer a usted mis escasos conocimientos, si es que pueden serle de alguna utilidad.

Aquiles: Oh, muchas gracias. Permítame llegar a la pregunta desde cierta persepectiva. ¿Conoce usted el grabado de M. C. Escher titulado *Cubo con Bandas Mágicas*?

Tortuga: ¿Donde hay cintas circulares con aplicaciones que parecen burbujas, y que ni bien se está seguro de que se trata de protuberancias, parecen transformarse en hendiduras, y viceversa?

Aquiles: El mismo.

Cangrejo: Lo recuerdo. Esas pequeñas burbujas parecen alternar constantemente entre presentarse como cóncavas y como convexas, según el punto desde el cual se las enfoque. No hay forma de verlas simultáneamente cóncavas convexas: de alguna manera, nuestro cerebro no nos lo permite. Hay dos "modos" recíprocamente excluyentes para la percepción de las burbujas.

Aquiles: Así es. Bien, creo haber descubierto dos modos análogos, en cierta forma, a través de los cuales yo puedo escuchar una fuga. Son éstos: uno, consiste en seguir una voz en particular por vez; el otro, en escuchar el efecto total del conjunto de voces, sin preocuparme por discernir entre ellas. Ya he probado con ambos modos y, para mi gran frustración, cada uno excluye al otro. Sencillamente, no estoy facultado para seguir el recorrido de las voces individuales y, al mismo tiempo, escuchar el efecto total. Descubro que me deslizo de un modo al otro, en forma más bien espontánea e involuntaria.

Oso Hormiguero: Tal como cuando se observan las bandas mágicas, ¿eh?

Aquiles: Sí. Me pregunto . . . si mi descripción de estos dos modos de escuchar las fugas no me marca irremisiblemente como un oyente ingenuo e inexperto, que ni siquiera puede comenzar a captar los modos más profundos de percepción que hay más allá del simple alcance de sus sentidos.

Tortuga: No, en absoluto, Aquiles. Sólo puedo hablar por mí misma, pero yo también me descubro alternando entre un modo y otro, fuera de todo control consciente acerca del modo que debe predominar. No sé si nuestros compañeros aquí presentes no han experimentado algo similar.

Cangrejo: Efectivamente. Más aun: es sin duda un fenómeno atormentador, pues uno siente que la esencia de la fuga le revolotea en torno, pero no puede aprehenderla por completo porque no le es posible hacer actuar ambos modos al mismo tiempo.

Oso Hormiguero: Las fugas tienen la interesante propiedad de que cada una de sus voces constituye, por sí misma, una composición musical; de tal modo, una fuga puede ser considerada una colección de diversas composiciones, basadas todas en un mismo y único tema, y ejecutadas todas simultáneamente. Y toca al oyente (o a su subconsciente) decidir si debe percibirla como una unidad, o como una colección de partes independientes que armonizan.

Aquiles: Usted dice que las partes son "independientes", pero eso no puede ser cierto, literalmente hablando. Tendrá que haber cierta coordinación entre ellas, pues de otro modo, al ser reunidas, se produciría una colisión nada sistemática de sonidos, y sabemos muy bien que no es esto lo que en verdad ocurre.

Oso Hormiguero: Una forma mejor de expresarlo puede ser la siguiente: si usted escucha cada voz por separado, hallará que parece tener sentido completo por sí misma. Podría existir en forma aislada, y es en este sentido que hablo de partes independientes. No obstante, tiene usted toda la razón cuando señala que cada una de estas líneas individualmente significativas se funde con las otras de un modo perfectamente planeado, lo cual genera una totalidad armoniosa. El arte de componer una bella fuga reside precisamente en la capacidad de elaborar varias líneas diferentes, cada una de las cuales crea la ilusión de haber sido compuesta en función de su belleza individual, pero que cuando son reunidas forman un conjunto, del cual no se tiene la impresión, en absoluto, de haber sido logrado forzadamente. Ahora bien, esta dicotomía entre la fuga escuchada como un todo, por un lado, y el seguimiento de las voces que la componen, por otro, es un caso particular de una dicotomía muy generalizada, presente en muchos tipos de estructuras construidas a partir de niveles inferiores.

Aquiles: Oh, ¿de veras? ¿Quiere usted decir que mis dos "modos" pueden ser aplicados con alguna mayor amplitud, a situaciones distintas a la de escuchar fugas?

Oso Hormiguero: Por cierto.

Aquiles: Estoy pensando cómo podrá ser eso. Supongo que tendrá relación con la alternancia entre percepción de conjunto de determinada cosa, y percepción de la misma como colección de partes. Pero la única vez en que me he topado hasta ahora con esta dicotomía ha sido escuchando fugas.

Tortuga: ¡Oh, pero, miren esto! Acababa de dar vuelta a la página, siguiendo la música, y me encuentro con esta magnífica ilustración, frente a la primera página de la fuga.

Cangrejo: No había visto nunca esa ilustración. ¿Podríamos verla?

(La Tortuga hace circular el libro. Los cuatro lo observan de manera peculiar: éste desde lejos, aquél desde muy cerca, todos moviendo la cabeza hacia uno y otro lado en señal de perplejidad. Cuando el libro vuelve a la Tortuga, ésta escudriña muy fijamente la página.)

Aquiles: Bueno, creo que el preludio está por terminar. Me pregunto si, cuando escuche la fuga, será más profunda mi comprensión de la pregunta, "¿Cuál es el modo correcto de escuchar una fuga: como un todo, o como la suma de sus partes?"

Tortuga: Escuche antentamente, ¡y usted mismo decidirá!

(*El preludio finaliza. Hay un momento de silencio, y . . .*

[ATTACCA]

CAPITULO X

Niveles de descripción y sistemas de computadora

Niveles de descripción

La cadena G de Gödel y una fuga de Bach comparten la propiedad de poder ser comprendidas en diferentes niveles. Todos estamos familiarizados con esta clase de cosas, pero en algunos casos caemos en confusiones, mientras que en otros nos las arreglamos sin ninguna dificultad. Por ejemplo, todos sabemos que los seres humanos están integrados por una enorme cantidad de células (veinticinco trillones, aproximadamente), y que, en consecuencia, todo lo que hacemos podría ser descripto, en principio, en función celular. O bien la descripción podría hacerse, inclusive, en el nivel de las moléculas. La mayoría de nosotros acepta esto de una manera bastante limitada; vamos al médico para que examine lo que consideramos nuestros niveles inferiores. Leemos sobre el ADN y la "ingeniería genética" y sorbemos nuestro café. Pareciera que hemos conciliado estas dos imágenes increíblemente distintas de nosotros mismos mediante el simple recurso de desconectarlas entre sí. No contamos prácticamente con ninguna forma de relacionar una descripción microscópica de nosotros mismos con lo que imaginamos que somos, y de ahí que sea posible el almacenamiento de representaciones disociadas de nosotros mismos en "compartimientos" de nuestra mente disociados por completo entre sí. Rara vez nos vemos en situación de alternar entre uno y otro de estos conceptos, preguntándonos, "¿Cómo estas dos cosas totalmente diferentes pueden ser el mismo *yo*?"

O bien, tomemos una secuencia de imágenes en una pantalla de televisión, las cuales muestran a Armando Manzanero sonriendo. Cuando observamos la secuencia, sabemos que en realidad no estamos viendo una persona, sino grupos de puntos titilantes sobre una superficie plana. Lo sabemos, pero es lo que tenemos más lejos de la mente. Contamos con estas dos representaciones diametralmente opuestas de lo que aparece en la pantalla, pero no nos confundimos. Podemos excluir una de ellas y pres-

tar atención a la otra: es lo que todos hacemos. ¿Cuál es "más real"? La respuesta depende de si se es un ser humano, un perro, una computadora o un aparato de televisión.

Bloques y pericia en ajedrez

Uno de los principales problemas dentro de las investigaciones dedicadas a las inteligencias artificiales consiste en resolver cómo se cierra la brecha existente entre aquellas dos descripciones: cómo construir un sistema que pueda reconocer un nivel de descripción, y producir el otro. Una de las formas a través de las cuales esta brecha penetró en el ámbito de las inteligencias artificiales es adecuadamente ilustrada por los avances registrados en la programación de computadoras para que jueguen ajedrez de alto nivel. Durante la década de los años cincuenta, y en los comienzos de la siguiente, se solía pensar que la clave para conseguir que la máquina jugase convenientemente residía en lograr que ésta previese, con mayor extensión que un maestro, la red ramificada de secuencias posibles del juego. Sin embargo, pese a que ese objetivo fue siendo alcanzado gradualmente, el nivel de juego de las computadoras no experimentó mejora súbita de ninguna clase, ni aventajó al de los seres humanos. Es un hecho que un jugador calificado puede derrotar, en forma totalmente segura y confiada, al mejor programa de la actualidad.

La razón que explica esto se encontraba publicada desde muchos años atrás. Durante los años cuarenta, el psicólogo Adriaan de Groot hizo estudios acerca de cómo perciben una situación de juego los principiantes y los maestros de ajedrez. Dicho en los términos más elementales, sus conclusiones sostienen que los maestros perciben la distribución de las piezas en *bloques*. Existe una descripción directa del tipo "blancas: P4R; negras: T6D"; a través de la primera es que el maestro produce, de alguna manera, su imagen mental del tablero. Esto fue demostrado por la gran rapidez con que un maestro puede reproducir una determinada posición tomada de una partida, en comparación con las dificultades que le presenta a un principiante la misma tarea, luego que ambos han observado el tablero durante cinco segundos. Fue altamente revelador el hecho de que las equivocaciones de los maestros afectaban la ubicación de *grupos* enteros de piezas, las cuales, pese a estar colocadas en casillas erróneas, componían una situación estratégica casi similar a la original; con los principiantes, no pasaba en absoluto lo mismo. La prueba decisiva consistió en repetir la experiencia, pero distribuyendo las piezas en forma arbitraria sobre el tablero, en lugar de disponerlas según una situación de juego. Se descubrió que los maestros no aventajaban a los principiantes en la reconstrucción de posiciones carentes de orden y concierto.

La conclusión es que en el ajedrez, normalmente, se reiteran ciertos tipos de situaciones —ciertos patrones—, y que la atención de los maestros

se dirige hacia estos patrones de alto nivel. Los pensamientos del maestro se desenvuelven *en un nivel distinto* al del principiante; su cuerpo de conceptos es diferente. Casi todo el mundo se asombra al descubrir que, durante una partida, rara vez un maestro va mucho más allá que un principiante en sus anticipaciones . . . ¡y por lo común, además, un maestro analiza solamente un puñado de movidas! Esto se explica porque su modo de percibir el tablero actúa como un filtro: cuando examina una situación de juego, literalmente *no ve las movidas inadecuadas*, tal como un simple aficionado no ve las movidas *ilegales* cuando hace su examen de situación. Nadie que haya jugado ajedrez, aun cuando su experiencia al respecto sea mínima, organiza su percepción del juego en función de movimientos diagonales de las torres, o de capturas hacia adelante por parte de los peones: estas variantes no ingresan jamás en su pensamiento. De modo análogo, los jugadores de nivel magistral han construido niveles más altos de organización en cuanto a la manera de observar el tablero; en consecuencia, es tan inverosímil que un jugador de esta índole dé entrada en su mente a una movida inadecuada, como que un aficionado corriente dé entrada en la suya a una movida ilegal. Esto podría ser llamado *poda implícita* de la gigantesca ramificación del árbol de posibilidades. Por contraste, la *poda explícita* abarcaría el análisis de una movida y luego de un estudio superficial la decisión de no profundizar el examen.

La distinción anterior puede aplicarse también a otras actividades intelectuales, como por ejemplo la matemática. Un matemático talentoso, por lo general, no estudia ni somete a prueba los recorridos falsos hacia el teorema que busca, mientras que la gente menos dotada se ve obligada a proceder así, trabajosamente; aquél se limita a "olfatear" los caminos prometedores y comienza a transitarlos de inmediato.

Los programas de ajedrez por computadora que se basan en la anticipación no han sido enseñados a pensar en un nivel más alto; la estrategia que han seguido se ha reducido al empleo de la fuerza bruta anticipatoria, esperando quebrantar así todo género de oposición. Sin embargo, no sirvió. Quizá algún día un programa de anticipación dotado de la suficiente fuerza bruta consiga vencer a los mejores jugadores humanos, pero ello no significará más que un modesto logro intelectual, si se lo compara con la revelación de que la inteligencia depende *decisivamente* de la capacidad de crear descripciones de alto nivel para aplicarlas a ordenamientos complejos, tales como los que presentan los tableros de ajedrez, las pantallas de televisión, las páginas impresas o las pinturas.

Niveles similares

Generalmente, no tenemos necesidad de retener más de un nivel de comprensión de una situación, por vez, en nuestra mente. Además, las distintas descripciones de un mismo sistema suelen estar tan alejadas,

conceptualmente, una de otra, que no hay inconveniente en mantenerlas separadas, como ya dijimos: se conservan en compartimientos mentales independientes. Sí hay motivo de confusiones cuando un mismo sistema admite dos o más descripciones que, a pesar de situarse en diferentes niveles, de alguna forma se *parecen* entre sí. En estos casos, se nos hace muy difícil evitar el entremezclamiento de los niveles cuando pensamos en tal sistema, y lo más probable es que se nos produzca una desorientación total.

Sin duda, nos sucede esto cuando reflexionamos acerca de nuestra propia psicología; cuando, pongamos por caso, intentamos comprender las motivaciones a que responden determinados actos. Hay muchos niveles en la estructura mental humana; se trata, por cierto, de un sistema que aún no conocemos muy bien. Así y todo, hay centenares de teorías que, cada una a su manera, explican por qué las personas actúan como lo hacen: todas estas teorías se basan en ciertos supuestos fundamentales acerca del grado de profundidad con que diversos tipos de "fuerzas" psicológicas se asientan en aquel conjunto de niveles.

Ahora bien, a causa de la utilización, en medida excesiva, de la misma clase de lenguaje para referirnos a todos los niveles mentales, resultan muchos entremezclamientos de niveles y, naturalmente, centenares de teorías erróneas. Por ejemplo, hablamos de "impulsos" —con respecto al sexo, al poder, a la fama, al amor, etcétera, etcétera— sin saber de qué parte de la estructura mental humana provienen. No voy a extenderme sobre este aspecto; sólo deseo agregar que nuestra confusión a propósito de lo que somos se relaciona, sin ninguna duda, con el hecho de que consistimos en un gran conjunto de niveles, y a que solemos utilizar un solo lenguaje para todos los niveles en función de los cuales nos describimos.

Sistemas de computadora

Hay otra esfera donde coexisten muchos niveles de descripción dentro de un mismo sistema, y donde todos los niveles se encuentran conceptualmente muy próximos entre sí. Hablo de los sistemas de computadora.* Un programa de computadora en funcionamiento puede ser visto a través de numerosos niveles. En cada uno de los niveles, la descripción está formulada en el lenguaje de la ciencia de las computadoras, por lo que todas las descripciones resultan, en cierto modo, similares entre sí, a despecho de las muy importantes diferencias que existen entre los criterios que uno se forma acerca de los distintos niveles. En el más bajo de éstos, la descrip-

* La palabra "computadora", y las expresiones que integra, carecen de la precisión que sería deseable. No obstante, el campo correspondiente a esta terminología, en español, presenta un exceso de variantes, ninguna de ellas enteramente satisfactoria. Hemos optado entonces por la utilización de "computadora", versión que cuenta con la ventaja de ser la más difundida y de enmarcar al lector con aceptable fidelidad en el orden de problemas que desarrolla el texto. [T.]

ción puede ser tan complicada que se asemeje a la descripción, en función de los puntos, de una imagen televisiva. Desde el punto de vista de determinados objetivos, empero, su gravitación es por demás sobresaliente. En el nivel más alto, la descripción se configura según grandes *bloques* y, pese a que comparte muchos conceptos con el nivel más bajo, adquiere un sentido totalmente diferente. Los bloques de esta descripción son como los de los ajedrecistas avezados, y como los de la descripción por grupos de la imagen que exhibe la pantalla: resumen, en forma de cápsula, cierta cantidad de cosas que en los niveles más bajos son tomadas por separado. (Véase la figura 57.)

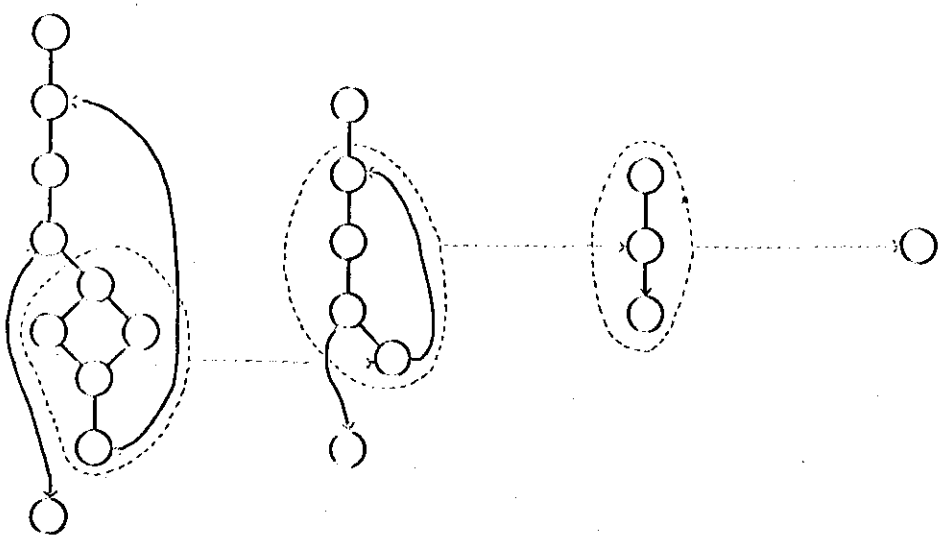
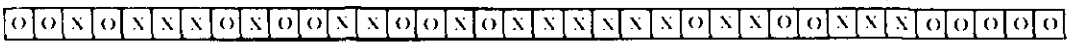


Figura 57. La noción de "bloque": un grupo de unidades es percibido de un modo distinto, ahora como un "bloque" individual. Las fronteras de un bloque se parecen un tanto a una membrana celular o a los límites de un país: establecen una identidad separada para el conjunto de dentro. De acuerdo al contexto, se prescindirá de la estructura interna del bloque, o bien se la tomará en consideración.

Ahora bien, antes de que las cosas se hagan demasiado abstractas, examinemos los hechos concretos relativos a las computadoras, comenzando por una rápida ojeada sobre cómo es un programa en su nivel más bajo. ¿Nivel más bajo? Bien, en verdad, no es así, pues no voy a hablar de las partículas elementales, pero en este momento no quiero hablar de ningún nivel más bajo que el de los programas.

En la base conceptual de una computadora, encontramos una *memoria*, una *unidad central de proceso* (UCP), y ciertos *dispositivos de*

entrada/salida (E/S). Primero, describiremos la memoria. Esta se divide en distintas unidades físicas, llamadas *palabras*. Para ser concretos, supongamos que haya 65536 palabras de memoria (un número típico, pues se trata de 2 elevado a la 16). La palabra es vuelta a subdividir en lo que podemos considerar los átomos de la ciencia de los computadores: los *bits*. El número de bits en una palabra común es de alrededor de treinta y seis. Desde el punto de vista físico, un bit no es más que un "interruptor", ubicado siempre en una de dos posiciones.



—una palabra de 36 bits—

Podemos identificar las dos posiciones como "arriba" y "abajo", o "x" y "o", o "1" y "0" . . . Esta última variante es la más usual: es perfectamente adecuada, pero puede tener el inconveniente de crear malentendidos, en la medida en que induzca a pensar que una computadora, en su base, almacena *números*. Y no es así. Hay tanto motivo para creer que un bit es un número como para creer que dos bits son el precio de un cucurucho de helado. Tal como el dinero puede obtener distintas cosas, según la forma en que se lo use, una palabra de memoria puede desempeñar una gran cantidad de funciones diversas. A veces, por cierto, los treinta y seis bits representarán un número en notación binaria. Otras, podrán representar treinta y seis puntos de una pantalla de televisión. Y otras más, podrán representar algunas letras de un texto. Cómo debe considerarse una palabra de memoria depende enteramente de la función que tenga asignada dentro del programa que la incluye. Por supuesto, puede cumplir más de una función, igual que una nota dentro de un canon.

Instrucciones y datos

Una de las interpretaciones posibles de una palabra, que aún no he mencionado, es la de *instrucción*. Las palabras de memoria no sólo contienen los datos con los cuales se debe operar, sino también el programa para operar con los datos. Existe un repertorio limitado de operaciones que pueden ser realizadas por la unidad central de proceso —la UCP—. Parte de una palabra, generalmente varios de sus primeros bits, es interpretable como el nombre del tipo de instrucción que debe aplicarse. ¿Qué papel cumple el resto de los bits de una palabra-interpretada-como-instrucción? Lo más frecuente es que indiquen qué otras palabras de la memoria han de ser operadas. En otros términos, los bits restantes constituyen un *señalador* de alguna otra palabra (o algunas otras palabras) de la memoria. Cada palabra tiene una ubicación propia en la memoria, lo

mismo que cada casa en una calle; tal ubicación es llamada la *dirección* de la palabra. La memoria, a su vez, puede tener una o muchas "calles", las cuales son llamadas "páginas". De tal modo, una palabra determinada obtiene su dirección mediante la cita de su número de página (si la memoria es paginada) y de su ubicación dentro de la página. De aquí que el aspecto de "señalamiento" de una instrucción consista en la dirección numérica de alguna(s) palabra(s) de la memoria. No hay restricciones para el señalamiento, por lo que una instrucción puede llegar a "señalarse" a sí misma; luego, cuando es ejecutada, ello es causa de que deba efectuarse una modificación en ella misma.

¿Cómo sabe la computadora qué instrucción debe ejecutar en un momento dado? De esto se encarga la UCP. Esta cuenta con un señalador especial que indica la siguiente palabra (es decir, suministra su dirección) a la que toca ser interpretada como una instrucción. La UCP extrae esa palabra de la memoria y la copia electrónicamente, lo cual da lugar a una palabra especial que pertenece específicamente a la UCP. (Las palabras de la UCP no son llamadas "palabras", sino *registros*.) Y entonces la UCP ejecuta esa instrucción. Ahora bien, una instrucción puede requerir que se realice cualquier operación, de entre una gran cantidad de tipos de éstas. Las más frecuentes incluyen:

SUMAR, a un registro, la palabra señalada por la instrucción.
(En este caso, la palabra señalada es, obviamente, interpretada como un número.)

IMPRIMIR, en forma de letras, la palabra señalada por la instrucción.
(En este caso, la palabra señalada es, obviamente, interpretada *no* como un número, sino *como* una cadena de letras.)

BRINCAR, hasta la palabra señalada por la instrucción.
(En este caso, se está indicando a la UCP que interprete esa palabra específica como la instrucción que sigue.)

A menos que la instrucción dicte explícitamente otra cosa, la UCP tomará la palabra ubicada inmediatamente a continuación, y la interpretará como una instrucción. En otros términos, la UCP da por supuesto que debe avanzar secuencialmente por la "calle", lo mismo que un cartero, interpretando palabra tras palabra como instrucciones. Pero este orden secuencial puede ser roto por instrucciones tales como la de BRINCAR* y otras.

* *Jump*, en el original: *brinco, salto*, y también *bifurcación*, en la terminología española de la especialidad y otras. [T.]

Lenguaje de máquina *vs.* lenguaje ensamblador

Este es un esbozo muy breve del *lenguaje de máquina*. Los tipos de operaciones que éste contiene constituyen un repertorio finito que no puede ser extendido. Así, todos los programas, por amplios y complejos que sean, deben elaborarse con componentes correspondientes a esos tipos de instrucciones. La observación de un programa formulado en lenguaje de máquina recuerda vagamente la observación de una molécula de ADN, presentada átomo por átomo. Si volvemos a echar un vistazo a la figura 41, la cual muestra la secuencia de nucleótidos de una molécula de ADN, y luego consideramos que cada nucleótido contiene aproximadamente dos docenas de átomos, y a continuación nos imaginamos que estamos tratando de enunciar el ADN, átomo por átomo, de un pequeño virus (¡no digamos de un ser humano!), tendremos, entonces, un atisbo de lo que es formular un programa complejo en lenguaje de máquina y de lo que es esforzarse por captar los logros de un programa cuando sólo se tiene acceso a su descripción en lenguaje de máquina.

Se debe recordar, sin embargo, que la programación de computadoras se hacía originalmente en un nivel aun más bajo, si cabe, que el del lenguaje de máquina: se interconectaban hilos conductores, creándose así un hardware para las operaciones requeridas. Esto es algo tan inauditamente primitivo desde la perspectiva de los usos actuales, que se hace trabajoso el solo imaginarlo. Pero es indudable que quienes lo hicieron por vez primera experimentaron el mismo alborozo que alguna vez llenó a los pioneros de las computadoras modernas . . .

Nos queremos desplazar ahora hasta un nivel más alto, dentro de la jerarquía de niveles de descripción de programas: el nivel del *lenguaje ensamblador*. No hay un espacio gigantesco entre el lenguaje ensamblador y el lenguaje de máquina; la distancia que va de uno a otro se transita más bien apaciblemente, por cierto. En sustancia, hay una correspondencia punto por punto entre instrucciones de lenguaje ensamblador e instrucciones de lenguaje de máquina. El propósito del lenguaje ensamblador es introducir "bloques" en las instrucciones específicas de un lenguaje de máquina, de modo que, en lugar de formular la secuencia de bits "010111000" cuando necesitamos una instrucción que sume un número a otro, simplemente escribimos SUMAR; luego, en lugar de suministrar la dirección en representación binaria, podemos referirnos a la palabra de memoria merced a un *nombre*.

En consecuencia, un programa formulado en lenguaje ensamblador es, en gran medida, algo así como un programa en lenguaje de máquina al que se ha hecho legible para los seres humanos. Podríamos comparar la versión en lenguaje de máquina de un programa con una derivación de TNT expresada en la oscura notación de la numeración Gödel; y a la versión en lenguaje ensamblador, con una derivación isomórfica de TNT, enunciada en la notación TNT original, la cual es mucho más fácil de

comprender. O bien, si retornamos a la imagen del ADN, podemos vincular la diferencia existente entre lenguaje de máquina y lenguaje ensamblador con la diferencia existente entre la laboriosa especificación de cada nucleótido, átomo por átomo, y la especificación de un nucleótido mediante el simple trámite de suministrar su *nombre* (esto es, 'A', 'G', 'C' o 'T'). Gracias a esta tan simple operación de introducción de "bloques", se produce un ahorro enorme de trabajo, pese a que las variaciones conceptuales no sean apreciables.

Programas que traducen programas

Es posible que lo más importante, con respecto al lenguaje ensamblador, no resida en las diferencias que lo separan del lenguaje de máquina, las cuales no son tan considerables, sino en la cuestión clave de que, a través suyo, los programas pueden ser enunciados en un nivel *absolutamente* distinto . . . Basta pensar en lo siguiente: el hardware está construido para "entender" programas en lenguaje de máquina —secuencias de bits— pero no letras ni números decimales. ¿Qué pasa si el hardware es alimentado con un programa enunciado en lenguaje ensamblador? Será como tratar de conseguir que una célula reconozca un trozo de papel donde se haya escrito la secuencia de nucleótidos utilizando símbolos alfabéticos en lugar de químicos. ¿Qué puede hacer una célula con un trozo de papel? ¿Qué puede hacer una computadora con un programa en lenguaje ensamblador?

Y aquí es donde aparece una circunstancia vital: es posible formular, en lenguaje de máquina, un *programa de traducción*. Este programa, llamado *ensamblador*, reconoce nombres de instrucciones mnemotécnicas, números decimales y otras abreviaturas adecuadas que un programador puede recordar con facilidad, y efectuar su conversión a monótonas pero precisas secuencias de bits. Una vez que el programa en lenguaje ensamblador ha sido *ensamblado* (es decir, traducido), es *procesado*: en realidad, lo que es procesado es su lenguaje de máquina equivalente; pero éste es un problema terminológico. ¿Qué nivel de programa está siendo procesado? Siempre será correcto decir que está en proceso el programa en lenguaje de máquina, pues siempre está involucrado el hardware cuando se procesa cualquier programa, pero también es enteramente razonable considerar el programa procesado bajo la forma de lenguaje ensamblador. Por ejemplo, se puede muy bien decir, "Ahora, la UCP ejecutará una instrucción de BIFURCACION", en lugar de, "Ahora, la UCP ejecutará una instrucción '11101000' ". Un pianista que ejecuta las notas Sol-Mi-Si Mi-Sol-Si también está ejecutando un arpegio en el acorde de Mi menor. No hay motivo para negarse a describir las cosas desde un punto de vista más elevado. Así, uno puede considerar que el programa en lenguaje ensamblador es procesado coexistentemente con el

programa en lenguaje de máquina. Tenemos dos modos de describir lo que está haciendo la UCP.

Lenguajes de nivel superior, compiladores e intérpretes

El siguiente nivel de la jerarquía lleva mucho más allá la idea, realmente poderosa, de traducir programas de nivel elevado a niveles más bajos. Luego de haberse utilizado la programación en lenguaje ensamblador durante bastantes años, durante el comienzo de la década de los cincuenta se advirtió que había una cantidad de estructuras características que reaparecían en todos los programas. Se diría que se trataba, como ocurre en el ajedrez, de ciertos patrones fundamentales que afloran en forma natural cuando los seres humanos procuran formular *algoritmos*: descripciones exactas de procesos que desean desarrollar. Dicho de otra manera, los algoritmos parecían tener ciertos componentes de nivel superior, aptos para especificar dichos procesos más fácil y elegantemente que a través del muy restringido lenguaje de máquina, o del lenguaje ensamblador. Habitualmente, un componente algorítmico de alto nivel consiste, no en una o dos instrucciones de lenguaje de máquina, sino en una colección completa de estas últimas, no necesariamente ubicadas en forma contigua en la memoria. Este componente puede ser representado, en un lenguaje de nivel superior, mediante una sola unidad: un bloque.

Además de los bloques así establecidos —esos componentes recién descubiertos, a partir de los cuales podían ser construidos todos los algoritmos— se comprobó que casi todos los programas contenían bloques aun más amplios: superbloques, por así decir. Estos difieren de programa a programa, según el tipo de tareas de alto nivel que toque cumplir al programa. Ya hablamos de superbloques en el Capítulo V, llamándolos de acuerdo a sus denominaciones habituales: "subrutinas" y "procedimientos". Estaba claro que el aditamento más eficaz, para cualquier lenguaje de programación, sería la capacidad de *definir* nuevas entidades de nivel más alto en función de las ya conocidas y luego *llamarlas* mediante un nombre. Esto supondría que la operación de elaborar bloques quedaría establecida dentro del lenguaje. En lugar de existir un repertorio determinado de instrucciones, a partir de las cuales tuviesen que ser ensamblados explícitamente todos los programas, el programador podría construir sus propios módulos, cada uno con su propio nombre, cada uno disponible para ser utilizado en cualquier punto del interior del programa, exactamente como si se tratara de un rasgo estructural del lenguaje. Por supuesto, no hay escapatoria con respecto al hecho de que, en el fondo, en el nivel del lenguaje de máquina, todo seguiría consistiendo en las mismas instrucciones en lenguaje de máquina de antes; no obstante, ello no estaría directamente a la vista del programador del alto nivel, sino que su presencia sería implícita.

Los nuevos lenguajes basados en las nociones anteriores fueron llamados *lenguajes compiladores*. Uno de los primeros y más precisos fue llamado "Algol", por abreviatura de "Algorithmic Language" (lenguaje algorítmico). A diferencia del caso del lenguaje ensamblador, no existe correspondencia directa, punto por punto, entre los enunciados del Algol y las instrucciones en lenguaje de máquina. Sin embargo, hay por cierto algún tipo de equivalencia entre el Algol y el lenguaje de máquina, pero es mucho más "embrollado" que el existente entre lenguaje ensamblador y lenguaje de máquina. Dicho esquemáticamente, un programa Algol es, a su traducción a lenguaje de máquina, lo que un problema verbal dentro de un texto elemental de álgebra es a la ecuación que lo traduce. (En verdad, obtener una ecuación a partir de un problema verbal es mucho más complejo, pero aporta una idea acerca de la clase de "desembrollamientos" que es necesario efectuar cuando se traduce de un lenguaje de alto nivel a un lenguaje de nivel más bajo.) A mitad de los años cincuenta, se consiguió formular programas denominados *compiladores*, encargados de realizar la traducción de lenguajes compiladores a lenguaje de máquina.

También fueron inventados los *intérpretes* o *interpretadores*. Igual que los compiladores, los intérpretes traducen de lenguajes de alto nivel a lenguaje de máquina, pero en lugar de traducir primero todos los enunciados y luego cumplir el código de la máquina, leen una línea y le dan ejecución de inmediato. Esto cuenta con la ventaja de que no se necesita haber formulado un programa completo para poder utilizar un intérprete. Se puede inventar el programa línea por línea, e irlo verificando a medida que se avanza. Así, la comparación entre un intérprete y un compilador es análoga a la que podemos establecer entre un intérprete simultáneo y un traductor de discursos escritos. Uno de los lenguajes de computadora más importante y también más fascinante es el LISP, abreviatura de "List Processing" (proceso de lista), creado por John McCarthy casi al mismo tiempo en que lo fue el Algol. A posteriori, el LISP se difundió grandemente entre los especialistas en inteligencias artificiales.

Hay una diferencia interesante entre el modo en que actúan, respectivamente, los intérpretes y los compiladores. Un compilador recibe entrada (un programa completo de Algol, por ejemplo), y produce salida (una larga secuencia de instrucciones en lenguaje de máquina). Llegado a este punto, el compilador ha cumplido su misión. La salida es entregada entonces a la computadora para su procesamiento. En cambio, el intérprete está procesando constantemente al tiempo que el programador va imprimiendo un enunciado LISP tras otro, cada uno de los cuales es ejecutado de inmediato. Pero esto no significa que cada enunciado es primero traducido y luego ejecutado, pues en tal caso un intérprete no sería otra cosa que un compilador línea por línea. Por el contrario, las operaciones de leer una nueva línea, "comprenderla" y ejecutarla están entrelazadas en el intérprete: suceden simultáneamente.

Esta es pues la concepción, sobre la que nos extenderemos un poco

más. Cada vez que es impresa una nueva línea de LISP, el intérprete trata de procesarla. Esto quiere decir que el intérprete entra en acción, y que determinadas instrucciones (en lenguaje de máquina), ubicadas en su interior, resultan ejecutadas. *Cuáles* resultan ejecutadas depende, precisamente, del propio enunciado LISP, como es lógico. Hay muchas instrucciones de BIFURCACION dentro del intérprete, de manera que una nueva línea de LISP puede ser causa de que el control se desplace en una forma compleja: hacia adelante, hacia atrás, luego otra vez hacia adelante, etc. Así, cada enunciado LISP llega a convertirse en un "sendero" en el interior del intérprete, y el seguimiento de tal sendero logra la obtención del efecto buscado.

En ocasiones, es útil considerar los enunciados LISP como simples secciones de datos, con los cuales es alimentado en orden secuencial un programa en lenguaje de máquina (el intérprete LISP), cuyo procesamiento es permanente. Cuando se conciben las cosas de este modo, se adquiere una imagen diferente de la relación que existe entre un programa formulado en un lenguaje de nivel más alto y la máquina que lo ejecuta.

Ganchos

Por supuesto que un compilador, por tratarse asimismo de un programa, debe ser formulado en algún lenguaje. Los primeros compiladores eran formulados en lenguaje ensamblador, y no en lenguaje de máquina, aprovechando así por entero el escalón ya conquistado por sobre el len-

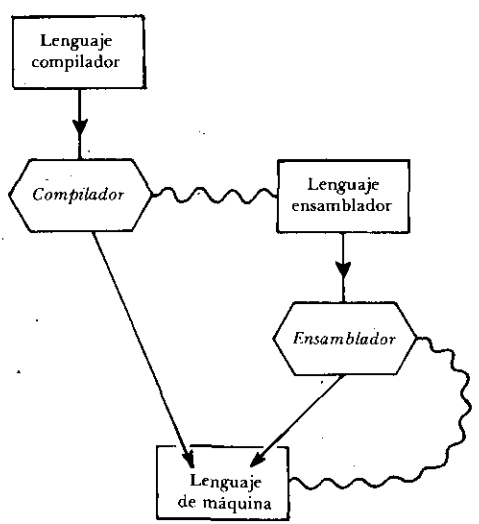


Figura 58. Tanto ensambladores como compiladores son traductores a lenguaje de máquina. Esto es indicado por las líneas rectas. Además, como también son programas, *son, a su vez, formulados originalmente en un lenguaje. Las líneas onduladas indican que un compilador puede ser formulado en lenguaje ensamblador, y un ensamblador en lenguaje de máquina.*

guaje de máquina. La figura 58 presenta una síntesis de estos complicados conceptos.

Ahora bien, al hacerse mayor aun el refinamiento, se advirtió que un compilador parcialmente formulado podía ser utilizado para compilar prolongaciones de sí mismo. En otras palabras, una vez que ha sido formulado un cierto núcleo mínimo de un compilador, ese compilador mínimo puede traducir compiladores mayores a lenguaje de máquina; éstos, por su parte, pueden traducir compiladores todavía mayores, hasta dejar compilado el compilador máximo. Este proceso es identificado gráficamente como del "gancho" o del "calzador", por razones obvias (al menos, para el lector cuyo idioma nativo sea el inglés).* No es muy distinto del proceso de adquisición de lenguaje por parte de un niño que ya alcanzó un nivel decisivo de fluidez en su idioma nativo: a partir de este punto, su vocabulario y fluidez pueden crecer a pasos agigantados, ya que puede *utilizar* el lenguaje para *adquirir* nuevo lenguaje.

Niveles para describir programas de procesamiento

Los lenguajes compiladores no reflejan, por lo general, la estructura de las máquinas que habrán de procesar los programas formulados en ellos. Esta es una de sus principales ventajas sobre el altamente especializado lenguaje ensamblador y sobre el lenguaje de máquina. Por supuesto, cuando un programa en lenguaje compilador es traducido a lenguaje de máquina, el programa que resulta es dependiente de la máquina. En consecuencia, un programa que está siendo ejecutado puede ser descripto de manera independiente de la máquina, o bien de manera dependiente de la máquina. Es como hacer referencia al parágrafo de un libro mencionando su tema (que es algo independiente del editor), o su número de página y localización dentro de la misma (que es algo dependiente del editor).

En la medida en que un programa está siendo pasado correctamente, no tiene mayor importancia el modo en que uno describa o conciba su funcionamiento. Pero cuando algo anda insatisfactoriamente, sí es importante ser capaz de pensar en diferentes niveles. Si, por ejemplo, se ha instruido a la máquina para que divida por cero en alguna etapa, aquélla hará un alto y permitirá que el usuario advierta el problema mediante la indicación del sitio del programa donde radica la dificultad. Sin embargo, esta especificación es suministrada, a menudo, en un nivel más bajo que el empleado por el programador para formular el programa. Siguen tres descripciones paralelas de un programa abocado a señalar un alto:

* "Gancho" y "calzador" son los términos que los especialistas manejan como equivalentes de "bootstrap", en español. Esta última es la palabra de resonancia obvia a que alude el texto: es el nombre de la tirilla o presilla, cuya forma semeja la de un gancho, ubicada en el extremo superior de las botas y que sirve para empujar a éstas hacia arriba en la operación de calzárselas. [T.]

Nivel de lenguaje de máquina:
"Ejecución del programa detenida en la posición 1110010101110111"

Nivel de lenguaje ensamblador:
"Ejecución del programa detenida al llegar a la instrucción DIV (dividir)"

Nivel de lenguaje compilador:
"Ejecución del programa detenida en el transcurso de la resolución de la expresión algebraica '(A + B)/Z'"

Uno de los mayores problemas que enfrentan los programadores de sistemas (o sea, las personas que formulan compiladores, intérpretes, ensambladores y otros programas, para que sean utilizados por muchas otras personas) es determinar cómo formular rutinas de detección de errores, de modo tal que sus mensajes al usuario, cuyo programa tiene un defecto, provean descripciones de alto nivel, antes bien que de bajo nivel, del problema. Como contraparte interesante, recordemos que cuando algo funciona mal en un "programa" genético (una mutación, por ejemplo), el defecto se manifiesta al observador humano únicamente en un nivel *alto*, a saber, en el nivel del fenotipo, no en el del genotipo. En realidad, la biología moderna ha hecho de las mutaciones una de sus principales ventanas a los procesos genéticos, a causa de la posibilidad que abren de efectuar rastreos en muchos niveles.

Microprogramación y sistemas operativos

En los sistemas modernos, la jerarquía incluye varios otros niveles. Algunos sistemas, por ejemplo, cuentan con instrucciones en lenguaje de máquina, las cuales son todavía más rudimentarias que la instrucción de adicionar un número de memoria a un número de un registro. Corresponde al usuario decidir qué clase de instrucciones en nivel corriente de máquina querría programar; luego, "microprograma" tales instrucciones bajo la forma de las "microinstrucciones" disponibles. Entonces, las instrucciones en "lenguaje de máquina de nivel superior" que el usuario haya previsto pueden ser incorporadas al diseño del circuito, e inclusive convertirse en hardware, aunque no sea imprescindible que así ocurra. La microprogramación permite al usuario la posibilidad de descender un tanto con respecto al nivel convencional de lenguaje de máquina. Una de las consecuencias de ello consiste en que una computadora de determinado fabricante pueda ser provista (a través de la microprogramación) de un hardware tal, que la haga contar con el mismo conjunto de instrucciones en lenguaje de máquina que una computadora distinta producida

por el mismo fabricante, o inclusive por otro. Se dice que la computadora microprogramada "imita" a otra.

Tenemos después el nivel del *sistema operativo*, situado entre el programa en lenguaje de máquina y cualquier nivel más alto que el usuario programa. El sistema operativo es de por sí un programa, cuya función es evitar el acceso de los usuarios a la máquina misma (protegiendo así el sistema), y también alejar al programador de los muchos y sumamente intrincados y dificultosos problemas de lectura del programa, para lo cual llama a un traductor, procesa el programa traducido, dirige la salida hacia los canales adecuados en el momento preciso y traslada el control al usuario siguiente. Si hay varios usuarios "hablándole" simultáneamente a la misma UCP, el sistema operativo es el programa que desplaza la atención hacia uno y otro de una manera ordenada. Las complejidades de los sistemas operativos son, ciertamente, formidables: me limitaré a insinuarlas, gracias a la analogía que sigue.

Consideremos el primer sistema telefónico: Alexander Graham Bell pudo comunicarse por teléfono con su ayudante, ubicado en el cuarto contiguo, ¡la transmisión electrónica de la voz! Esto equivale a una computadora simple, sin sistema operativo: ¡la computación electrónica! Veamos ahora un sistema telefónico moderno. Es posible elegir otros teléfonos para comunicarse con ellos; no sólo eso, sino que pueden procesarse muchas llamadas al mismo tiempo. Se puede anteponer un prefijo y conectarse así con diferentes áreas. Se puede llamar directamente, a través del operador, por cobrar, a pagar mediante carta de crédito, persona a persona. Se pueden invertir los pasos de una llamada, o determinar su fuente. Existe una señal de teléfono ocupado. Existe una señal con sonido de sirena para indicar que el número discado no está "bien formado", o que hemos demorado excesivamente en discar. Existen conmutadores que permiten conectar a un grupo de teléfonos situados en un lugar determinado . . . etcétera, etcétera. Esta lista es asombrosa, si se considera la flexibilidad que presenta, en comparación con el otrora milagroso teléfono "desnudo".

Ahora bien, los refinados sistemas operativos realizan análogas operaciones de manejo del tráfico y de desplazamiento de niveles con respecto a los usuarios y sus programas.

Es prácticamente indudable que hay acá cierto paralelo con fenómenos propios del cerebro: manejo de muchos estímulos simultáneos; decisiones acerca de prioridades y plazo de mantenimiento de éstas; "interrupciones" momentáneas causadas por emergencias u otros hechos inesperados; y así siguiendo.

Amortiguadores para el usuario y protección para el sistema

La gran cantidad de niveles de un sistema complejo de computadora se combinan para "amortiguar" al usuario, evitándole tener que pensar en

el funcionamiento de los muchos niveles inferiores, los cuales, es lo más probable, carecen de toda importancia para él, de cualquier manera. Por lo general, un pasajero de avión no necesita preocuparse por el nivel de combustible de los tanques, o por la velocidad de los vientos, o por la cantidad de pollo que ha de servirse para la cena, o por la situación del tráfico aéreo alrededor del punto de destino: todo ello es dejado en manos del personal ubicado en los distintos niveles de la jerarquía de la aerolínea; el pasajero se limita a viajar entre un lugar y otro. Y también aquí ocurre que, sólo cuando algo funciona *erróneamente* —como, por ejemplo, que su equipaje no haya llegado—, el pasajero se entera del complicado sistema de niveles que se extiende debajo suyo.

¿Las computadoras son superflexibles o superrígidas?

Uno de los objetivos principales, en la búsqueda de niveles más altos, ha sido siempre el de tratar de convertir en algo lo más natural posible la tarea de comunicar a la computadora qué se requiere de ella. Sin duda, las construcciones de alto nivel en lenguajes compiladores están mucho más cerca de los conceptos que los seres humanos piensan naturalmente, en comparación con las construcciones de bajo nivel, tales como las del lenguaje de máquina. En esta tendencia hacia la sencillez de la comunicación, sin embargo, uno de los aspectos de la "naturalidad" ha sido desatendido por completo. Se trata del hecho de que la comunicación interhumana tiene constricciones mucho menos rígidas que la comunicación entre seres humanos y máquinas. Por ejemplo, cuando perseguimos la forma más adecuada de expresar algo, frecuentemente producimos fragmentos de oraciones que carecen de sentido, carraspeamos en mitad de las frases, nos interrumpimos recíprocamente con nuestros interlocutores, utilizamos descripciones ambiguas y formas sintácticas incorrectas, inventamos expresiones y distorsionamos significados . . . pero así y todo nuestro mensaje consigue completarse en su mayor parte. Con los lenguajes de programación, se ha seguido generalmente la regla de establecer una sintaxis sumamente estricta, la cual ha de ser aplicada de modo riguroso; no hay palabras ni construcciones ambiguas. Es interesante el hecho de que esté permitido el equivalente impreso del carraspeo (es decir, un comentario inesencial o carente de relevancia), pero únicamente si se lo ha establecido en forma previa mediante una clave (por ejemplo, la palabra COMENTARIO), a cuyo final debe ir, como cierre, otra palabra clave (un punto y coma, digamos). Esta leve inclinación a la flexibilidad contiene, irónicamente, su propia trampa: si es utilizado un punto y coma (o cualquier palabra clave señalada para finalizar un comentario) en el interior de un comentario, el programa de traducción interpretará que dicho punto y coma está indicando el fin del comentario y cercenará lo que sigue.

Si ha sido definido un procedimiento denominado COMPRENSION,

y luego llamado diecisiete veces por el programa, pero cuando es llamado por decimoctava vez se lo deletrea equivocadamente como COMPRESNION, ¡ay del programador! El compilador se plantará, e imprimirá un rígido y severo mensaje de error, diciendo que jamás oyó hablar de COMPRESNION. A menudo, cuando es detectado un error semejante por un compilador, éste trata de continuar, pero a causa de su falta de compresnión no puede entender qué es lo que quiere decir el programador. En realidad, puede muy bien dar por supuesto que se ha querido decir algo completamente diferente y proceder sobre la base de tal interpretación errónea; en tal caso, una larga serie de mensajes de error habrán de condimentar el resto del programa, pues el compilador —no el programador— se ha confundido. Imaginemos el caos que se produciría si un intérprete simultáneo de inglés-ruso, luego de escuchar una expresión francesa insertada en un discurso en inglés, se pusiera a tratar de interpretar el resto íntegro del discurso inglés como si fuese francés. Los compiladores suelen extraviarse de esta dramática manera. *C'est la vie.*

Quizá esto parezca una impugnación a las computadoras, pero no es así. En algún sentido, las cosas tienen que desenvolverse de esta forma. Si uno se detiene a pensar por qué tanta gente utiliza computadoras, comprende que es para que realicen tareas definidas y precisas, caracterizadas por el hecho de ser excesivamente complejas para que se aboquen a ellas las personas. Para que sea posible confiar en la computadora, es imprescindible que ésta comprenda, sin que se deslice ni la más remota ambigüedad, qué se espera de ella. Es necesario, también, que no haga ni más ni menos que lo establecido explícitamente por las instrucciones. Si la amortiguación ubicada por debajo del programador incluye un programa cuyo objeto sea "conjeturar" qué desea o qué quiere decir el programador, es muy probable que el programador sea totalmente malinterpretado cuando pretenda comunicar en qué consiste su tarea. En consecuencia, si bien resulta cómoda para el ser humano, la programación de alto nivel deberá ser precisa e inequívoca.

Anticiparse a las conjeturas del usuario

Ahora bien, es posible idear un lenguaje de programación —y un programa que lo traduzca a niveles inferiores— que permita cierto género de imprecisiones. Una forma de plantear esto sería decir que un traductor de aquel lenguaje de programación tratará de encontrarle sentido a elementos que han sido elaborados "fuera de las reglas del lenguaje". Sin embargo, si un lenguaje permite algunas "transgresiones", ¡éstas ya no lo son cabalmente, puesto que han sido incluidas en las reglas! Si un programador sabe que puede cometer ciertos errores de deletreo, le es posible entonces hacer uso deliberado de aquel rasgo del lenguaje, pues sabe que está operando, en realidad, dentro de las rígidas reglas del mismo, pese a las apa-

riencias. En otros términos, si el usuario está enterado de todas las flexibilidades programadas, para su conveniencia, en el traductor, conoce entonces los límites que no puede rebasar y, en consecuencia, el traductor se le aparecerá todavía como algo rígido e inflexible, a pesar de que le permite una libertad mucho mayor que las primeras versiones de lenguaje, las que carecían de "corrección automática de errores humanos".

Frente a lenguajes "elásticos" de este tipo, habría una alternativa: 1) el usuario está enterado de las flexibilidades incorporadas al lenguaje y a su traductor; 2) el usuario no está enterado de las mismas. En el primer caso, el lenguaje mantiene su aptitud para comunicar programas con precisión, ya que el programador puede predecir cómo va a interpretar la computadora los programas que él formula en el lenguaje. En el segundo caso, la "amortiguación" cuenta con aspectos ocultos que pueden dar lugar a cosas inesperadas (para el particular punto de vista de un usuario que desconoce el funcionamiento interior del traductor). Esto puede resultar en graves malinterpretaciones de programas, de modo que tal lenguaje se hace inadecuado con relación a objetivos que necesitan principalmente rapidez y seguridad por parte de la computadora.

En verdad, hay una tercera opción: 3) el usuario está enterado de las flexibilidades incorporadas al lenguaje y a su traductor, pero éstas son tantas e interactúan de una manera tan compleja, que no se puede determinar cómo serán interpretados los programas. Esto puede ser muy bien ilustrado por el caso de la persona que formuló el programa de traducción: indudablemente, conoce el contenido del programa mejor que nadie, pero no le es posible anticipar cómo habrá de reaccionar frente a ciertas construcciones inusuales.

Una de las áreas sobresalientes de investigación en la actualidad, en materia de inteligencias artificiales, es la conocida como *programación automática*, la cual se interesa por el desarrollo de los lenguajes de más alto nivel que existen; lenguajes cuyos traductores son muy refinados, al punto de poder realizar, cuando menos, algunas de estas impresionantes tareas: generalizar a partir de ejemplos; corregir ciertos errores gramaticales o de impresión; probar de otorgarle sentido a descripciones ambiguas; tratar de adelantarse a las conjeturas del usuario mediante el empleo de un modelo de usuario original; hacer preguntas frente a cuestiones que no están claras; usar el idioma en sus expresiones, etc. Lo que uno espera es poder recorrer la cuerda floja tendida entre precisión y flexibilidad.

Los progresos en materia de inteligencias artificiales son progresos en materia de lenguajes

Es sorprendente lo directo de la relación que existe entre los progresos que obtiene la ciencia de las computadoras (en especial, lo referente a inteligencias artificiales) y el desarrollo de nuevos lenguajes. En la década

anterior surgió una nítida tendencia: consolidar los descubrimientos relativos a nuevos lenguajes. Una clave para la comprensión y la creación de inteligencias reside en el desarrollo y refinamiento constantes de los lenguajes encargados de describir los procesos de manipulación simbólica. Hay en la actualidad alrededor de tres o cuatro docenas de lenguajes experimentales, aparecidos como consecuencia, exclusivamente, de las investigaciones sobre inteligencias artificiales. Es importante percatarse de que cualquiera de los programas que pueden ser formulados en uno de estos lenguajes es programable, en principio, en lenguajes de nivel más bajo, pero ello requeriría un esfuerzo tremendo por parte del ser humano y, además, el programa resultante sería tan extenso que excedería los alcances de la aprehensión humana. No se trata de que todo nivel más alto amplíe el potencial de la computadora: todo este potencial ya está presente en su conjunto de instrucciones en lenguaje de máquina. Se trata de que los nuevos conceptos contenidos en un lenguaje de alto nivel sugieren, por su misma naturaleza, orientaciones y perspectivas.

El "espacio" de todos los programas posibles es tan gigantesco que nadie puede hacerse una idea al respecto. Todo lenguaje de nivel más alto deriva, naturalmente, de la exploración de ciertas regiones del "espacio de programa"; de tal modo, el programador, a través de la utilización de ese lenguaje, es orientado hacia aquellas áreas del espacio de programa. El lenguaje no lo *obliga* a formular programas de determinado tipo, sino que le hace más *fácil* la realización de ciertas tareas. La proximidad con respecto a un concepto, más un suave empujón, es todo lo que suele hacer falta para arribar a un descubrimiento mayor: y éste es el motivo que anima a la búsqueda de lenguajes dotados de niveles cada vez más altos.

Programar en diferentes lenguajes es como componer en diferentes tonalidades, en especial si se trabaja en el teclado. Para quien aprendió o compuso piezas en muchas tonalidades, cada tonalidad tendrá su propia y especial resonancia emotiva. Asimismo, cierta clase de figuraciones se le hará presente como si surgiese por sí sola en una tonalidad, pero en otra no le resultará tan fácil. De manera, pues, que la elección de tonalidad imprime una orientación. En algunos casos, inclusive tonalidades inarmónicas, como podrían serlo las de Do sostenido y Re bemol, provocan sentimientos enteramente distintos. Esto muestra la manera en que un sistema notacional puede desempeñar un papel significativo en el moldeamiento del resultado final.

La figura 59 exhibe una representación "estratificada" de inteligencia artificial, en la cual encontramos partes de máquina, tales como transistores, en la base, y "programas inteligentes" en la cúspide. Esta ilustración ha sido tomada del libro *Artificial Intelligence*, de Patrick Henry Winston, y trasunta una concepción compartida por casi todos los especialistas en inteligencias artificiales. Si bien estoy de acuerdo con la idea de que la inteligencia artificial debe ser estratificada de una manera semejante a ésta, no creo que, luego de tan pocos estratos, se puedan alcan-

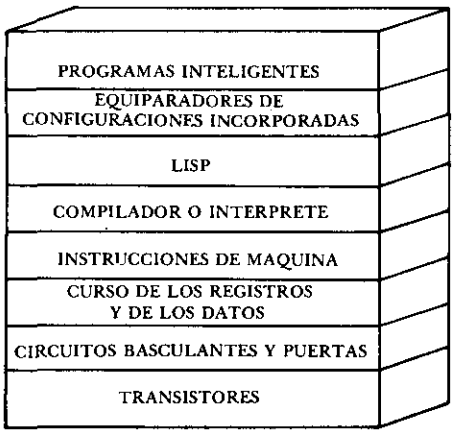


Figura 59. Para crear programas inteligentes, es necesario construir una serie de niveles de hardware y de software, a fin de evitar la angustia de verlo todo únicamente en el nivel más bajo. Las descripciones en distintos niveles de un mismo proceso darán la impresión de ser muy diferentes entre sí, pero sólo cuando el nivel superior esté lo suficientemente articulado en bloques como para que nos sea comprensible. [Adaptado de P. H. Winston, Artificial Intelligence *(Reading, Mass.: Addison-Wesley, 1977).]*

zar los programas inteligentes. Estoy convencido de que, entre el nivel del lenguaje de máquina y el nivel donde se llegue a la inteligencia misma, tiene que haber quizá una docena (¡o, inclusive, varias docenas!) más de estratos, donde cada uno de éstos se apoye en las flexibilidades del anterior y las extienda. En qué consistirán es algo que difícilmente podamos imaginar ahora . . .

El paranoico y el sistema operativo

La similitud que existe entre todos los niveles de un sistema de computadora puede provocar algunas extrañas experiencias de confusión de niveles. Tuve una vez la oportunidad de observar a dos de mis amigos —ambos en su etapa de aprendizaje— jugando, en una terminal, con el programa "PARRY". PARRY es un programa, bastante desacreditado, que imita a un paranoico de manera sumamente rudimentaria mediante el recurso de espetar expresiones grabadas, en inglés, que incluyen un amplio repertorio. El interés de este programa obedece a su capacidad de indicar cuáles, de entre las expresiones almacenadas, pueden parecer razonables para responder a las frases que, en inglés, le formula por escrito un ser humano.

En una ocasión, el tiempo de respuesta fue excesivo —PARRY se tomaba demasiado tiempo para responder— y expliqué entonces a mis amigos que ello se debía, probablemente, a las pesadas exigencias que soportaba el sistema de tiempo compartido. Les dije que ellos podían saber cuántos usuarios estaban contribuyendo a ello si imprimían un carácter especial, de "control", el cual se dirigiría directamente al sistema operativo, sin ser advertido por PARRY. Uno de mis amigos oprimió el carácter de control. Instantáneamente, aparecieron en la pantalla ciertos datos internos, relativos a la situación del sistema operativo, desplazando algunas

de las palabras de PARRY. PARRY no sabía nada de esto: es un programa que sólo "sabe" de carreras de caballos y de apuestas, no de sistemas operativos, ni de terminales, ni de caracteres especiales de control. Para mis amigos, sin embargo, PARRY y el sistema operativo eran, los dos, simplemente "la computadora": una entidad misteriosa, remota, amorfa, que les respondía cada vez que ellos imprimían una interrogación. Y así, les pareció perfectamente razonable que uno de ellos imprimiera idiomáticamente, muy suelto de cuerpo, la pregunta, "¿Por qué borra usted lo que está impreso en la pantalla?" La noción de que PARRY pudiera carecer de todo conocimiento acerca del sistema operativo que lo estaba procesando no había sido bien captada por mis amigos. La idea de que "usted" lo sabe todo acerca de "sí mismo" es tan corriente, en virtud del contacto con las demás personas, que fue natural extenderla a la computadora . . . ¡al fin y al cabo, ésta era lo suficientemente inteligente como para "hablarles" en inglés! La pregunta de mis amigos no se distinguió mucho de preguntarle a una persona, "¿Por qué elaboró tan pocos glóbulos rojos hoy?" La gente no tiene conocimiento de ese nivel —el "nivel del sistema operativo"— de su cuerpo.

La causa principal de esta confusión de niveles radicó en que la comunicación con todos los niveles tenía lugar a través de una sola pantalla, en una misma terminal. Pese a que la simpleza de mis amigos pueda parecer bastante extrema, ocurre a menudo que personas avezadas en el manejo de computadoras cometan errores similares, cuando diversos niveles de un sistema complejo concurren a la vez en la misma pantalla: olvidan a "quién" se están dirigiendo, e imprimen cosas que no tienen sentido en el nivel en el cual se ubican, pero que sí lo tendrían en otro nivel. Parecería deseable, en consecuencia, que el sistema mismo seleccione los niveles, a fin de interpretar las órdenes que "tengan sentido". Lamentablemente, tal interpretación requeriría que el sistema contase con una enormidad de sentido común, además de un perfecto conocimiento de todos los propósitos del programador: para ello, se necesita más inteligencia artificial de la que existe en la actualidad.

La frontera entre software y hardware

También puede crear confusión la flexibilidad de algunos niveles y la rigidez de otros. En algunas computadoras, por ejemplo, hay sistemas prodigiosos de compaginación de textos, los cuales hacen posible que distintas secciones de textos sean "vertidas" de un formato a otro, casi del modo en que un líquido es vertido de un recipiente a otro. Una página estrecha puede convertirse en una página ancha y viceversa. Ante ello, se podría esperar que fuese igualmente sencillo pasar de un tipo de letra a otro: digamos, de romanas a *itálicas*. Sin embargo, puede que no haya sino un tipo disponible en la pantalla, de modo que tales cambios sean

impracticables. O bien, podría ser que la pantalla los permitiese pero que la impresora no los pudiese imprimir o a la inversa. Luego de que uno se ha dedicado a las computadoras durante mucho tiempo, se torna exigente y piensa que todo es programable: ninguna impresora, entonces, debería ser tan rígida que no tenga más que un solo juego de caracteres, o inclusive un repertorio finito de éstos . . . ¡los tipos de letras deberían ser especificados por el usuario! Ahora bien, una vez obtenido ese grado de flexibilidad, uno puede sentirse incomodado porque la impresora no puede imprimir en diferentes colores, o porque no admite papeles de todas las formas y tamaños, o porque no se repara a sí misma cuando sufre un desperfecto . . .

La dificultad reside en que, en alguna parte, toda esta flexibilidad tiene que "sanearse", para usar la expresión del Capítulo V. Deberá existir un nivel de hardware que subyazca a todo y que sea inflexible. Podrá estar ubicado en profundidades recónditas, y haber tanta flexibilidad en los niveles de más arriba, que pocos usuarios lleguen a advertir las limitaciones del hardware, pero éste, inevitablemente, permanecerá allí.

¿En qué consiste la clásica distinción entre *software* y *hardware*? Es la distinción entre programas y máquinas: entre largas y complicadas secuencias de instrucciones y las máquinas físicas que las cumplen. Me gusta considerar al software como "todo aquello que puede ser transmitido más allá de las líneas telefónicas", y al hardware como "toda otra cosa". Un piano es hardware, pero la música impresa es software. Un aparato telefónico es hardware, pero un número telefónico es software. La distinción es provechosa pero no siempre tan tajante.

Los seres humanos también tenemos aspectos "software" y "hardware", y la diferencia entre unos y otros constituye una segunda naturaleza en nosotros. Estamos habituados a la rigidez de nuestra fisiología: no podemos curar nuestras enfermedades con sólo desearlo, o conseguir que el cabello nos crezca del color que se nos ocurra, para mencionar únicamente un par de ejemplos sencillos. Sin embargo, podemos "reprogramar" nuestras mentes a fin de que operen dentro de nuevos marcos conceptuales. La extraordinaria flexibilidad de nuestras mentes parece casi irreconciliable con la noción de que nuestros cerebros deben estar compuestos por reglas fijas de hardware, las cuales no pueden ser reprogramadas. No podemos conseguir que nuestras neuronas se exciten con mayor o menor rapidez, no podemos rehacer los circuitos de nuestros cerebros, no podemos rediseñar el interior de una neurona, no podemos ejercitar *ninguna* opción con respecto al hardware . . . Sin embargo, podemos controlar la manera en que pensamos.

Pero hay, sin duda, aspectos del pensamiento que están más allá de nuestro control. No podemos hacernos más talentosos mediante un acto de voluntad; no podemos aprender un idioma nuevo con la rapidez que querríamos; no podemos, por nosotros mismos, agregar velocidad a nuestros pensamientos; no podemos, por nosotros mismos, llegar a pensar

en varias cosas al mismo *tiempo*; y así siguiendo.

Este es un género de autoconocimiento primordial cuyo carácter obvio, precisamente, lo hace difícil de percibir: algo parecido a lo que sucede con el aire que nos rodea. Jamás nos preocupamos, realmente, por reflexionar acerca de lo que puede ser la causa de esas "imperfecciones" de nuestra mente, a saber: la organización de nuestro cerebro. Proponer formas de reconciliación entre el software de la mente y el hardware del cerebro es uno de los objetivos principales de este libro.

Los niveles intermedios y el clima

Hemos visto que en los sistemas de computadora existen varios estratos bastante rigurosamente definidos; la operación de un programa en proceso puede ser descripta en función de cualquiera de ellos. No hay, pues, un único nivel bajo y un único nivel alto, sino toda una graduación, tanto hacia un extremo como hacia el otro. La existencia de niveles intermedios, ¿es un rasgo común de los sistemas que cuentan con bajo y alto nivel? Consideremos, por ejemplo, el sistema cuyo "hardware" es la atmósfera de la tierra (no tan "hard" —duro—, pero no importa), y cuyo "software" es el clima. Seguir el movimiento simultáneo de todas las moléculas sería una forma de muy bajo nivel de "comprender" el clima, y no la consideración de un enorme y complicado programa en el nivel del lenguaje de máquina. Obviamente, esa forma sobrepasa las posibilidades de captación humana. Sin embargo, conservamos nuestras propias formas, peculiarmente humanas, de observar y describir los fenómenos climáticos. Nuestra visión, articulada en bloques, del clima, está basada en fenómenos de muy alto nivel, tales como: lluvia, niebla, nevada, huracanes, frentes fríos, estaciones, presiones, alisios, la corriente de chorro, nubes de lluvia, tormentas eléctricas, capas de inversión térmica, etc. Todos estos fenómenos involucran una cantidad astronómica de moléculas, las cuales se conciertan de alguna manera para conseguir que emerjan los efectos de gran escala. Esto tiene algún parecido con la consideración del clima en un lenguaje compilador.

¿Existe algo que tenga analogía con la consideración del clima en un lenguaje de nivel intermedio, tal como un lenguaje ensamblador? ¿Hay, pongamos por caso, pequeñas "minitormentas" locales, cosas como los reducidos torbellinos que suelen verse, arremolinando polvo en columnas de muy escasos metros de diámetro? Una ráfaga de viento en un sitio determinado, ¿es un bloque de nivel intermedio que cumple un papel en la generación de fenómenos climáticos de nivel más alto? ¿O no hay, sencillamente, formas prácticas para combinar el conocimiento de tales tipos de fenómenos en una explicación más amplia del clima?

Se me ocurren otras dos preguntas. La primera es: "¿Sería posible que los fenómenos climáticos que percibimos en nuestra escala —un tornado,

una sequía — fuesen tan sólo fenómenos de nivel intermedio: partes de fenómenos más vastos y despaciosos?" Si es así, entonces los verdaderos fenómenos climáticos de alto nivel serían globales, y su escala temporal la geológica. La edad del hielo sería un acontecimiento climático de alto nivel. La segunda pregunta es: "¿Hay fenómenos climáticos de nivel intermedio que hayan escapado por completo a la percepción humana, pero que, en caso de ser percibidos, aportarían un conocimiento mucho mayor acerca de por qué el clima es como es?"

De los tornados a los quarks

Esta última sugerencia puede parecer arbitraria, pero no lo es tanto. Nos basta con observar la más rigurosa de las ciencias rigurosas —la física— para encontrar singulares ejemplos de sistemas que son explicados en función de la interacción de "partes", las cuales son invisibles. En la física, como en cualquier otra disciplina, un *sistema* es un grupo de *partes* que interactúan. En la mayoría de los sistemas que conocemos, las partes retienen su identidad en el transcurso de la interacción, de modo que podemos seguir viéndolas en el interior del sistema. Por ejemplo, cuando se reúne un equipo de futbol en asamblea, los jugadores retienen su individualidad: no la disuelven en alguna clase de entidad compuesta, la cual confundiría las particularidades personales. Sin embargo —y esto es importante— en sus cerebros se ponen en marcha ciertos procesos, los cuales son desencadenados por el contexto colectivo y que de otra manera no tendrían lugar; así, en escala menor, los jugadores modifican su identidad cuando se convierten en parte del sistema mayor, el equipo. Este tipo de sistema es llamado *sistema cercano a la descomponibilidad* (noción que proviene del artículo de H. A. Simon, *The Architecture of Complexity*; véase la Bibliografía). Tal sistema consiste en módulos débilmente interactivos, cada uno de los cuales conserva su identidad propia durante la interacción; sin embargo, experimentan una leve diferenciación con respecto a cómo son fuera del sistema, lo que les permite contribuir al funcionamiento cohesionado de todo aquél. Los sistemas estudiados por la física son, generalmente, de esta clase. Se considera que un átomo, pongamos por caso, está formado por un núcleo cuya carga positiva captura cierta cantidad de electrones, a los que mantiene en "órbitas", o estados ligados, alrededor suyo. Estos electrones ligados son sumamente semejantes a los electrones libres, pese a su carácter de elementos internos de un objeto compuesto.

Algunos de los sistemas estudiados por la física ofrecen un contraste en relación con el relativamente sencillo átomo. Un ejemplo lo da el núcleo del átomo, el cual suele ser descripto como "un conjunto de protones y neutrones". Pero las fuerzas que compelen a las partículas componentes a mantenerse juntas son tan podersosas, que dichas partículas no sobrevi-

ven a través de ninguna manera que se asemeje a su forma "libre" (la forma que presentan cuando no pertenecen a un núcleo). En realidad, además, un núcleo actúa, en muchos aspectos, como una partícula individual y no como un conjunto de partículas en interacción. Cuando un núcleo es fragmentado, los protones y los neutrones quedan a menudo en libertad, pero también ocurre que son producidas otras partículas, por lo general, tales como mesones pi y rayos gamma. ¿Estaban físicamente presentes estas diferentes partículas en el interior del núcleo, antes de ser éste fragmentado, o sólo son "chispas" que se desprenden con la fragmentación? Procurar una respuesta a tal pregunta quizá no sea importante. En el nivel de la física de las partículas, no es muy clara la diferencia existente entre almacenar el potencial que produzca "chispas" y almacenar subpartículas concretas.

Un núcleo es, así, un sistema cuyas "partes", aun cuando no sean visibles mientras permanecen en el interior de aquél, pueden ser extraídas y puestas a la vista. Y hay otros casos patológicos: los del protón y el neutrón vistos a su vez como sistemas. Se supone de ambos, por vía de hipótesis, que están constituidos por una terna de "quarks": partículas hipotéticas que pueden ser combinadas de a dos, o de a tres, formando de tal modo muchas partículas fundamentales. Sin embargo, la interacción entre quarks es tan fuerte que, además de no poder ser vistos en el interior del protón y del neutrón, ¡no pueden ser extraídos de allí de ninguna manera! En consecuencia, aunque los quarks contribuyen a la comprensión teórica de ciertas propiedades de los protones y de los neutrones, es posible que nunca se pueda establecer su propia existencia independiente. Tenemos aquí, entonces, la antítesis de un "sistema cercano a la descomponibilidad": o sea, un sistema del cual, si algo se puede señalar, es su carácter "cercano a la indescomponibilidad". Con todo, es curioso que las teorías sobre protones y neutrones (y otras partículas) basadas en los quarks cuentan con una notable capacidad esclarecedora, mostrada por el hecho de que gran cantidad de resultados experimentales relativos a las partículas de las cuales se da por supuesto que están constituidas por quarks pueden ser perfectamente explicados, de modo cuantitativo, mediante el empleo del "modelo quark".

La superconductividad: una "paradoja" de renormalización

En el Capítulo V hablamos de la manera en que las partículas renormalizadas surgían de sus centros desnudos, merced a interacciones recursivamente elaboradas con partículas virtuales. Una partícula renormalizada puede ser vista así, como compleja construcción matemática, o bien como el gránulo individual que, físicamente, es. Una de las más extrañas y dramáticas consecuencias que se derivan de esa forma de describir las

partículas es la explicación que proporciona acerca del célebre fenómenos de la *superconductividad*: el flujo de electrones, libre de resistencia, en ciertos sólidos sometidos a temperaturas extremadamente bajas.

Resulta que esos electrones son renormalizados a través de sus interacciones con curiosos quanta de vibración llamados *fonones* (¡también ellos renormalizados!). Estos electrones renormalizados son llamados *polarones*. Los cálculos muestran que, a temperaturas muy bajas, dos polarones que rotan en sentido opuesto comienzan a atraerse, y pueden muy bien llegar a unirse de determinada manera. Bajo condiciones adecuadas, todos los polarones que transportan corriente se aparearán, formando *pares de Cooper*. Lo paradójico es que tal apareamiento sucede, precisamente, porque los electrones —los centros desnudos de los polarones apareados— se repelen eléctricamente entre sí. En contraste con los electrones, ningún par de Cooper experimenta atracción ni repulsión hacia cualquier otro par; por consiguiente, pueden deslizarse libremente a través de un metal, como si éste fuese un vacío. Si la descripción matemática de tal metal, donde las unidades originales sean polarones, es transformada en otra, donde esas unidades sean pares de Cooper, se obtendrá un conjunto considerablemente simplificado de ecuaciones. Esta simplicidad matemática es el modo a través del cual los físicos saben que la articulación en "bloques", representada por los pares de Cooper, es la perspectiva natural para estudiar la superconductividad.

Tenemos aquí varios niveles de partículas: el par de Cooper mismo; los dos polarones en rotación opuesta que lo componen; los electrones y fonones que dan lugar a los polarones, y después, los fotones y positrones virtuales en el interior de los electrones, etc. Podemos observar cada nivel y percibir fenómenos en él, que son explicados por el conocimiento que se haya adquirido acerca de los niveles anteriores.

"Tabicamiento"

Análogamente, y por suerte, no es necesario saberlo todo a propósito de los quarks para comprender muchas cosas relativas a las partículas que éstos componen. Así, un físico nuclear puede manejarse con teorías de los núcleos que se basen en protones y neutrones, e ignorar las teorías de los quarks y las que compiten con éstas. El físico nuclear tiene una imagen articulada en *bloques* de los protones y los neutrones: una descripción que deriva de teorías de nivel más bajo, pero que no requiere el conocimiento de las teorías de ese nivel. Del mismo modo, un físico atómico tiene una imagen articulada en bloques de un núcleo atómico, derivada de la teoría nuclear. Y un químico tiene una imagen articulada en bloques de los electrones y sus órbitas, y elabora teorías sobre las moléculas pequeñas, teorías que pueden ser asumidas, articulándolas en bloques, por los biólogos moleculares, quienes tienen una idea a propósito de cómo se

reúnen las moléculas pequeñas, pero cuya experiencia específica tiene lugar en el campo de las moléculas sumamente grandes y el modo en que éstas interactúan. Y los biólogos celulares tienen una imagen articulada en bloques de las unidades estudiadas por los biólogos moleculares, a las que tratan de emplear para establecer cómo interactúan las células. El asunto está claro. Todo nivel, en cierto sentido, está "tabicado" con respecto a los niveles inferiores. La anterior es otra de las expresiones gráficas de Simon, alusiva a la construcción en compartimientos de los submarinos, de manera tal que si una parte es dañada, y el agua comienza a filtrarse, el peligro puede ser conjurado cerrando las puertas; separando, es decir, tabicando, el compartimiento dañado con respecto a los compartimientos vecinos.

Aunque siempre hay "goteras" dentro de los niveles jerárquicos de la ciencia, ya que un químico no puede permitirse una ignorancia total con respecto a los niveles más bajos de la física, ni un biólogo lo mismo con respecto a la química, casi no hay goteras entre niveles distantes uno de otro. Es por ello que la gente puede comprender intuitivamente a otra gente sin necesidad de comprender el modelo quark, la estructura de los núcleos, la naturaleza de la órbita de los electrones, la afinidad química, la estructura de las proteínas, los organelos de la célula, los métodos de comunicación intercelular, la fisiología de los distintos órganos del cuerpo humano, o las complejas interacciones existentes entre los órganos. Todo lo que necesita una persona es un modelo articulado en bloques de cómo actúa el nivel más alto; por lo que sabemos, tales modelos son muy veraces y logrados.

El regateo entre articulación en bloques y determinismo

Sin embargo, quizá haya un importante aspecto negativo en el modelo de bloques: carece, por lo general, de la capacidad de predecir con exactitud. Esto es, mediante el empleo de ese tipo de modelo eludimos la tarea imposible de considerar a la gente como conjuntos de quarks (o lo que fuere que existe en el nivel más bajo), pero el modelo de bloques, por supuesto, nos proporciona solamente estimaciones probabilísticas sobre lo que sienten otras personas, sobre cómo reaccionarán ante cosas que digamos o hagamos, y así por el estilo. En síntesis, al usar modelos de bloques de alto nivel, estamos sacrificando el determinismo a la simplicidad. Aunque no estemos seguros de la forma en que reaccionará la gente ante una broma, igual la hacemos, con la expectativa de que la respuesta será reírse o no reírse: en lugar de, digamos, ponerse a trepar la columna de alumbrado más cercana (¡un maestro zen sí podría hacer esto último!). Un modelo de bloques define un "espacio", dentro del cual se espera que se ubique la conducta, y especifica las probabilidades de ubicación en las diferentes partes de ese espacio.

"Las computadoras sólo pueden hacer lo que se les indica que hagan"

Ahora bien, estas ideas pueden ser tan provechosamente aplicadas a los programas de computadoras como a los sistemas físicos compuestos. Hay un viejo dicho que reza, "Las computadoras sólo pueden hacer lo que se les indica que hagan". Esto es correcto en un sentido, pero pasa por alto la siguiente circunstancia: no se conocen por adelantado las consecuencias de lo que se le dice a la computadora que haga; luego, la conducta de ésta puede ser tan desconcertante, sorprendente e imprevisible como la de una persona. Por lo común, se conoce con anticipación el *espacio* dentro del cual se ubicará la salida, pero no se sabe detalladamente cuál será el punto de ubicación. Por ejemplo, es posible formular un programa destinado a calcular el primer millón de dígitos de π. El programa descerrajará dígitos de , hasta el final, mucho más rápido de lo que nosotros podríamos hacerlo: pero no es paradójico que la computadora deje atrás al programador. El espacio donde se ubicará el resultado es conocido con anterioridad: es el espacio que cubren los dígitos de 0 a 9; es decir que se cuenta con un modelo de bloques del comportamiento del programa. Pero si se conociera el resto, no se habría formulado el programa.

Hay otro sentido en el cual este viejo dicho también yerra. Se trata del hecho de que, cuando se programa en niveles cada vez más altos, ¡cada vez se sabe menos qué se le ha indicado a la computadora que haga! Estratos y estratos de traducción pueden separar el propósito inicial de un programa complejo de las instrucciones efectivas en lenguaje de máquina. En el nivel en el que se piensa y programa, los enunciados pueden parecer declarativos y tener el aspecto de sugerencias, más que de enunciados imperativos u órdenes. Y toda la trepidación interna que provoca la entrada de un enunciado de alto nivel nos pasa, por lo común, desapercibida, lo mismo que, cuando comemos un emparedado, no nos tomamos el trabajo de tomar conciencia del proceso digestivo que se ha desencadenado.

De todos modos, esta noción de que "las computadoras sólo pueden hacer lo que se les indica que hagan", formulada originalmente por Lady Lovelace en sus célebres memorias, se ha generalizado tanto y está tan asociada a la idea de que "las computadoras no pueden pensar", que volveremos a ella en el último capítulo, cuando nuestros niveles de apreciación hayan crecido en refinamiento.

Dos tipos de sistema

Hay una distinción importante que divide en dos tipos diferentes los sistemas constituidos por muchas partes. Hay sistemas donde el comportamiento de algunas partes tiende a *invalidar* el comportamiento de otras partes, con el resultado de que pierde importancia lo que ocurre en el ba-

jo nivel, porque casi todos los elementos, en el alto nivel, no diferencian entre sí sus respectivos comportamientos. Un ejemplo de esta clase de sistema lo ofrece un recipiente que contenga gas, donde todas las moléculas se entrechocan y detonan en formas microscópicas muy complejas; pero el resultado global, desde un punto de vista macroscópico, es un sistema sumamente calmo y estable, dotado de determinada temperatura, presión y volumen.

Y hay sistemas donde el efecto de un solo hecho en el nivel bajo puede ser *magnificado* a través de una resonancia enorme en el nivel alto. Un ejemplo de tal sistema lo encontramos en una máquina de *pinball*, donde el ángulo exacto en el que una bola golpee cada poste es fundamental para determinar el resto de su curso descendente.

Una computadora es una elaborada combinación de los dos sistemas. Contiene subunidades tales como hilos metálicos, cuyo comportamiento es altamente predictible: conducen electricidad ajustándose a la ley de Ohm, que es una ley muy precisa, articulada en bloques, parecida a las leyes que gobiernan los gases contenidos en el recipiente, puesto que depende de los efectos estadísticos resultantes de billones de efectos azarosos que se invalidan entre sí, lo cual genera un comportamiento total predictible. Una computadora también contiene subunidades macroscópicas, tales como una impresora, cuyo comportamiento está enteramente determinado por delicados patrones de intensidades eléctricas. Lo que resulta impreso de ninguna manera emana de una miríada de efectos microscópicos de cancelación. En realidad, en el caso de la mayor parte de los programas de computadoras, el valor de cada bit individual del programa cumple una función decisiva en la salida que es impresa. Si cualquier bit es modificado, también la salida se modifica radicalmente.

Los sistemas formados exclusivamente por subsistemas "seguros" —es decir, subsistemas cuyo comportamiento pueda ser predicho con seguridad, gracias a descripciones por bloques— juegan un papel de importancia inestimable en nuestra vida cotidiana, pues son garantías de estabilidad. Podemos confiar en que las paredes no se vengan abajo, en que las calles se dirijan adonde lo hacían ayer, en que el sol alumbre, en que los relojes indiquen la hora correctamente, etc. Los modelos de bloques correspondientes a esos sistemas son, virtualmente, por completo deterministas. Por supuesto, la otra clase de sistema que gravita de modo significativo en nuestra vida es la dotada de un comportamiento variable, el cual depende de ciertos parámetros microscópicos internos —cuya cantidad suele ser muy grande, además— que eluden nuestra observación directa. Nuestro modelo de bloques de un sistema así estará formulado necesariamente en función del "espacio" de operación, y comprenderá estimaciones probabilísticas acerca de las diferentes regiones de ese espacio donde pueda producirse el desembarco.

Un contenedor de gas que, como ya puntualicé, es un sistema confiable a causa de los múltiples efectos de cancelación que se producen, obedece

a rigurosas y deterministas leyes de la física. Son *leyes articuladas en bloques*, en la medida en que consideran el gas como un conjunto e ignoran sus constituyentes. Asimismo, la descripción microscópica y la macroscópica de un gas utilizan términos totalmente diferentes. La primera requiere que se especifiquen la posición y la velocidad de cada una de las moléculas componentes; la segunda requiere únicamente la especificación de tres nuevos fenómenos: temperatura, presión y volumen, los dos primeros de los cuales ni siquiera tienen equivalentes microscópicos. La simple relación matemática que vincula estos tres parámetros —$pV = cT$, donde c es una constante— es una ley supeditada a los fenómenos de nivel más bajo, pero es independiente de éstos. Por otra parte, es una ley que permite ignorar por completo el nivel más bajo, si así se quiere; es en este sentido que es independiente del nivel más bajo.

Es importante entender que la ley de alto nivel no puede ser formulada utilizando el vocabulario de la descripción de bajo nivel. "Presión" y "temperatura" son términos nuevos, que no pueden ser alcanzados a través de la sola experiencia con el nivel bajo. Los seres humanos podemos percibir directamente la temperatura y la presión; esto se explica por el modo en que estamos construidos, por lo cual no es sorprendente que hayamos podido descubrir esta ley. Pero las criaturas que solamente conozcan los gases como construcciones matemáticas teóricas deberían contar con la capacidad de sintetizar conceptos nuevos para poder idear esta ley.

Epifenómenos

Para cerrar este capítulo, me gustaría narrar un cuento relativo a un sistema complejo. Conversaba yo un día con dos programadores de sistemas de la computadora que estaba usando. Decían ellos que el sistema operativo se mostraba capaz de arreglarse para satisfacer con gran comodidad a cerca de treinta y cinco usuarios, pero que a partir de ese número, poco más o menos, el tiempo de respuesta se dilataba súbitamente, llegando a ser tan lento que uno podía hacer el registro y luego irse a su casa a esperar. En broma, dije, "¡Bueno, esto es fácil de solucionar: basta con ubicar el sitio del sistema operativo donde está almacenado el número '35', y cambiarlo por '60'!" Festejaron mi ocurrencia. La gracia reside, por supuesto, en que tal sitio no existe. ¿Dónde aparece, entonces, el número crítico: 35 usuarios? La respuesta es: *Es una consecuencia visible de toda la organización del sistema: un "epifenómeno"*.

Lo mismo sería preguntarle a un atleta, "¿Dónde está almacenado el '11' que lo hace a usted capaz de correr 100 metros en 11 segundos?" Obviamente, en ninguna parte. Esa marca es resultado de cómo está construido el corredor, de cuál es su tiempo de reacción y de un millón más de factores, todos en interacción cuando aquél corre. La marca es perfectamente reproducible, pero no está almacenada en ninguna parte de su

cuerpo. Está diseminada en todas las células de su organismo y sólo se manifiesta a través de la carrera misma.

Los epifenómenos abundan. En el juego del "go", existe la situación en que "subsisten dos ojos". No está construida por las reglas, pero es una consecuencia de las reglas. En el cerebro humano hay credulidad. ¿Cuán crédulo es uno? ¿La credulidad se localiza en algún "centro de la credulidad", dentro del cerebro? ¿Un neurocirujano podría ubicarlo y realizar alguna suerte de complicada operación que haga decrecer la credulidad, u optar por dejarlo en paz? Si el lector cree que la respuesta a lo anterior es afirmativa, ello revela que es bastante crédulo, y que quizá debería pensar en someterse a tal operación.

Mente *vs.* cerebro

En los capítulos venideros nos explayaremos acerca del cerebro; examinaremos entonces si el nivel superior del cerebro —la mente— puede ser comprendido sin necesidad de comprender los niveles más bajos de los cuales, a la vez, depende y no depende. ¿Hay leyes del pensamiento que estén "tabicadas" con respecto a las leyes de más abajo que gobiernan la actividad microscópica de las células del cerebro? ¿La mente puede ser "rebanada" del cerebro y trasplantada a otros sistemas? ¿O bien es imposible discernir subsistemas nítidos y modulares dentro de los procesos del pensamiento? ¿El cerebro es semejante a un átomo, a un electrón renormalizado, a un núcleo, a un neutrón o a un quark? ¿La conciencia es un epifenómeno? Para comprender la mente, ¿es necesario hacer todo el recorrido descendente hasta el nivel de las células nerviosas?

Figura 60. Adaptación del dibujo del autor. [MU = MU; ¡HOLISM = HOLISMO!; REDUCTIONISM = REDUCCIONISMO.]