Here is again the same parse of the Information that paragraph from *Strong and Efficient Cache Side-Channel Protection Using Hardware Transactional Memory* (2017 USENIX).

	Circ.1	Circ.2	Participant 1	Pro-	Circ.3	-cess	Participant 2	Circ.4	Circ.5
1			HTM			allows for	the efficient implementation of parallel algorithms		
2			it	is	commonly	used to elide	expensive software synchronization mechanisms		
3	informally	for a CPU thread executing a hardware transaction	all other threads			appear to be halted			
4		from the outside	a transaction			appears as	an atomic operation		
5			a transaction			fails		if the CPU cannot provide this atomicity	due to resource limitations or conflicting concurrent memory accesses
6			all transactional changes			need to be rolled back			

7		To be able to detect conflicts and revert transactions	the CPU			needs to keep track of	transactional memory accesses		
8			transactional memory	is	typically	divided into	a read set and a write set		
9			A transaction's read set			contains	all read memory locations		
10			Concurrent read accesses by other threads to the read set	are	generally	allowed			
11			concurrent writes			are	problematic		
11'	depending on the actual HTM implementation and circumstances	likely	[concurrent writes]			lead to	transactional aborts		
12			any concurrent accesses to the write set		necessarily	lead to	a transactional abort		

Once more, what you are seeing here are the units of information. In Part 10, I parsed this paragraph to draw your attention to the Process and to the Participants. I told you: Process and Participant stand at the center of every clause. The follow-on to this fact is that Circumstances stand around the center. Really, there's no surprise in this — I mean, that is literally the etymology of the word: Latin *cirum* "around" + Latin *stare* "stand." Thus, a Circumstance is peripheral. However, peripheral does not mean unimportant. In fact, in the context of the system

Given-New, where the reader is gauging levels of informativeness in the words on the line — in this situation, it turns out that the periphery can prove to be the most important place for information to be located.

Position is highly functionalized in the clause. For example, the thematic front position makes those words into the topic of the clause, and writers need topics to decide what to say next. That's how the system of Theme-Rheme works. Well, in the system of Given-New, the situation is pretty much turned right around: The front position of a clause will normally hold, from the reader's viewpoint, the information which is given, while the back positions in the clause will normally hold the information which is new.

Let me show you what I mean. Here's our example paragraph once more, but this time with just one unit highlighted in each clause.

	Circ.1	**Circ.2**	**Participant 1**	**Pro-**	**Circ.3**	**-cess**	**Participant 2**	**Circ.4**	**Circ.5**
1			HTM			allows for	**the efficient implementation of parallel algorithms**		
2			it	is	commonly	used to elide	**expensive software synchronization mechanisms**		
3	informally	for a CPU thread executing a hardware transaction	all other threads			**appear to be halted**			

4		from the outside	a transaction			appears as	**an atomic operation**		
5			a transaction			fails		if the CPU cannot provide this atomicity	**due to resource limitations or conflicting concurrent memory accesses**
6			all transactional changes			**need to be rolled back**			
7		To be able to detect conflicts and revert transactions	the CPU			needs to keep track of	**transactional memory accesses**		
8			transactional memory	is	typically	divided into	**a read set and a write set**		
9			A transaction's read set			contains	**all read memory locations**		
10			Concurrent read accesses by other threads to the read set	are	generally	**allowed**			
11			concurrent writes			are	**problematic**		
11'	depending on the actual HTM implementation and circumstances	likely	[concurrent writes]			lead to	**transactional aborts**		
12			any concurrent accesses to the write set		necessarily	lead to	**a transactional abort**		

The pattern here will be obvious, but to be sure: In each clause, the final unit is the one that provides the information which will be newest to the intended reader.

For example, just take the particularly complex clause No.5. That clause ends on two long Circumstances, but each of these is clearly newer to the reader than the information at the center of the clause (i.e., the Participant *a transaction* and the Process *fails*). Moreover, the information of Circ.5 is clearly newer than the information of Circ.4. How can I be so sure? Well, for one, the word *this* in Circ.4 is a dead giveaway. In scientific prose, the word *this* points backward. Consequently, the word serves the function of retrieving from a previous clause (in this case, clause No.4) information now needed in the current clause. Circ.5, in contrast, introduces *accessing* into the discourse. This is the reader's first encounter with this operation which, by becoming now more given to the reader, can then make the informational background to the clauses now to come. In short, Circ.5 in clause No.5 is such an important unit of information that it sets the course for the main point of this paragraph, namely, how HTM can run in parallel.

In essence, the system Given-New is parsed in reverse to the system Theme-Rheme. The units colored red in the above parse have the heaviest informational value, and this is a quite typical structuring of the information in adjoining clauses. As a reader moves to the back of each clause, the informational values become heavier and heavier. Actually, it is possible to assign rough weights to the informational values of clausal material. A workable approach (i.e., not oversimple and not overcomplicated) is to assign one of three different weight-

values to each unit of a clause. First, we let the light value equal W1. This value will, by default, overlap with the Theme, that is, with the clausal position most associated with background information. Next, we let the heavy value equal W2, and for W2, the default position is in the Rheme. However, the value W2 never extends to the full Rheme, and the reason is that the system of Given-New imposes this restriction on every clause of every text: **One information unit must carry the value of W3.**

The restriction actually arises from readers' needs. A reader must have something very new in each clause, otherwise the reader will stop reading. The analogy you might draw here is to a conversation. If the person you are speaking to begins repeating an anecdote you've heard, you'll interrupt saying, "Yeah, you told me already." Well, it's not different in text: The reader demands of every clause something else, something more or something new. Consequently, the rhematic positions in a clause will normally be divided between heavy and heavier weight-values, or in our notation here, between W2 and W3 information units. However, the rhematic position that really counts and the one which must be occupied is the one reserved for W3. In fact, an entire clause may by composed of just this weight-value W3 on its own and by itself. That is why W3 actually has a name, and that name is the Focus. The Focus is the reason the clause exists, at least as the reader sees the clause.

Right, now let me illustrate a bit how these three weight-values appear in clauses. Here's my parse of the informational values of clause No.5 above.

	W1	W2	W3
5	a transaction fails	if the CPU cannot provide this atomicity	due to resource limitations or conflicting concurrent memory accesses

Or, to provide another point of comparison, here's my parse of clause No.7 above:

	W1	W2	W3
7	To be able to detect conflicts and revert transactions, the CPU	needs to keep track of	transactional memory accesses

Or, more complex than these, here is my parse of the dual clause No.11 above:

	W1	W2	W3
11 and 11'	concurrent writes	are depending on the actual HTM implementation and circumstances likely lead to	problematic transactional aborts

Notice how it is the punctuation that actually wants to control the Information of the clause. By eliding the Participant *concurrent writes*, the two Circumstances (1) *depending on the actual HTM implementation* and circumstances and (2) *likely* can be weighted heavier at W2. Because, remember, weight of Information tends to

lighten as a unit advances to the front of a clause. Therefore, in the dual clause No.11, the punctation (i.e., the dash) and the complex structuring (i.e., the elision) increase the weight of the front positions.

The key takeaway here is this: Given-New decides how the reader understands a text, while Theme-Rheme decides how the writer makes the text understandable. That's why the systems operate in opposing directions, but together.

In the next post, I'll continue to explore these interrelations, because these are the interrelations which determine how Information comes across in text.