Some Simple Algebraic Properties of Merges

It should be simple enough to consider a hierarchy of combinatorial operations ranked from the most general to the most constrained. There is some suggestion that a more general operation is simpler and has greater expressive power. This is either empty rhetoric or false. The fact that a system has many ways to produce the same structure does not render it more expressive. Expressivity is a matter of weak generation. If a system can produce many structures in many different ways, then its ambiguity can render it useless. An ambiguous sentence might 'express' more but it says less (under any reasonable analysis of informational or semantic content). Similarly, we shouldn't mistake an increase in weak expressive power for strong expressive power. Given a set of strings S, a system which gives us only one way to generate each string is empirically more powerful than one which gives several derivations.

Splurge: commutative, associative
Altmerge: commutative, alternative
Merge: commutative, non-associative[1]
Concatenation: non-commutative, associative
Tree-adjunction: non-commutative, non-associative

To these we can add:

Adgerge: commutative, non-associative, external operation
Stablerge: commutative, non-associative, internal function

Three points first:
i) It is important not to be mislead when thinking about commutativity. The axiom $xy = yx$ has nothing whatsoever to do with linear order. Commutativity is a mathematical property that doesn't concern our notation. $1 + 2 = 2 + 1$ not because of our writing system allow us to state them in two orders but because they both are equal to 3. That is, 3 can be analysed as $1+2$ or $2+1$ under the addition operation. This is a fact about combinatorics, not linear representations. The fact that concatenation 'recognises' linear order while merge does not should not be mistaken with our orthographic conventions.

ii) Altmerge involves the alternative property of operations.
left-alternative: $x + (x + y) = (x + x) + y$
right-alternative: $(y + x) + x = y + (x + x)$

[1] Chomsky 2007 writes "Another complication beyond pure Merge is adding the principles of associativity and ordering, suppressing hierarchy and yielding sequences." He has also in lectures suggested that concatenation is more complex than merge due to its lack of commutativity.

So Altmerge(x, {x,y}) = {{x},y} = {y,{x}}. There might be some interest in this for the derivation of functional projections through external merge rather than self-merge. I won't look at this now.

iii) Clark (2011) has shown that non-associativity is not a necessary condition for the representation of syntactic structure. A Complete Idempotent Semi-ring provides a natural semantic interpretation
for context-free grammars despite the fact that both of its operations ○ and ⌄ are associative. This important observation suggest that a (two level) hierarchy of associative operations can accomplish work done otherwise by assuming non-associativity as the basis of hierarchical structure.

Adgerge and Stablerge are attempts to limit the powers of merge by placing constraints on its domain of application. The aim is to ensure that the system does not have access to the powerset of lexical items when it is selecting elements to merge. In both cases merge applies within a 'workspace' and only to items which are within the range of a select operation. Stablerge restricts the power of merge by making a workspace dynamic. The workspace is represented as a set and with each successive merge the set changes. Under Stablerge, you never merge in the same workspace twice. The dynamic account of the workspace, W, is provided in the definition : For some lexical items A, B, $A \in W_i$, either A contains B or Wi immediately contains B, and $W_i+1 = (W_i - \{A, B\}) \cup \{Merge(A,B)\}$. This rules out merging occurring between subsets of elements (parallel merge - {x, y}, {y, z} = {(x, (y), z)})since in this case the merged elements are both subsets of previous workspaces.[2]

Adgerge keeps the workspace static but stipulates that there are two of them; an operating space (OS) and a resource space (RS). Merge only occurs within the domain of OS which has a maximum cardinality of 2, but elements can be selected from either OS or RS. The RS has access to the lexicon (long term memory) and the operating space. It is a multiset (to allow repetitions), it stores the resources for continuing derivations, and halting is defined relative to it - a computation halts when it has one element in it.3

The neat idea is that while RS is a multiset, it does not itself contain multisets. In fact it cannot contain multisets because any element of it which isn't a lexical item is a set and to have become a set it would have to have undergone the operation of binary merge with another element. As a result, no subset can immediately contain more than two elements and even if they were the same element when they were merged in OS e.g.

---

[2] The odd notation here is attempting to show 'y's position in the intersection of two previous merges.

[3] Adgerge is technically simpler but theoretically more complex because it concerns the relationship between set-theoretic and multiset-theoretic structures.

merge(x, x), this self-merge would have produced a singleton {x} rather than {x, x} because OS only recognises sets, not multisets.

This means that merge cannot occur between two subsets of previously merged elements and so cases of parallel merge, head-movement and roll-up are blocked.
In both cases merge is binary. Adgerge is an external binary operation while Stablerge is an internal binary function. An internal binary operation is a mapping from $A \times A \rightarrow A$, i.e. its domain and range are both A. Since A is a set and we don't want merge to have access to P(A) but we do want it to have access to past outputs of A. For external merge to occur it must be immediately preceded by a selection of an item from RS. As such, external merge can be seen as a mapping : $A \times B \rightarrow A$ where A is OS and B is RS. The fact that OS is a set and RS is a multiset constrains the possible outputs of merge. In contrast, Stablerge is an internal binary function, or a mapping from one set to another $A \times A \rightarrow B$, i.e. $W_i \times W_i \rightarrow W_i{+}1$.