Appendix A

Glossary

This dissertation defines many new terms and uses some existing terms in a non-standard way. As an aid to the reader, this glossary contains definitions of these terms. Each entry includes a bracketed number that indicates a page on which the term is either introduced or discussed.

across [132] An orientation for a tile operation indicating that the operation is unordered with respect to subcall initiations and returns. Applies to both linear and tree-shaped tiles. Also used to describe the shape of a sliver exhibiting no inter-layer data dependencies.

aggregate data approach [82] An approach to the signal processing style of programming in which the signal processing operators are represented as procedures that manipulate aggregate data structures (e.g., lists, arrays, streams, trees).

arguments [131] The values consumed by a procedure application. In tile diagrams, they appear as inputs terminating on the line representing the call boundary.

attribute grammar [124] A grammar-based formalism for declaratively specifying the decoration of a tree with attributes.

barrion [192] A pair of a down synchron and an up synchron that represents a call boundary.

barrier synchronization [302] A style of synchronization in which all members of a group of processes must rendezvous at a point before any are allowed to proceed. Synchrons provide barrier synchronization for a dynamically determined number of processes.

between [148] An orientation for a tree tile operator indicating that the operator must be performed after some subcall return but before another subcall initiation.

between-LR [148] An orientation for a binary tile operator indicating that the operator must be performed after the left subcall return but before the right subcall initiation.

between-RL [148] An orientation for a binary tile operator indicating that the operator must be performed after the right subcall return but before the left subcall initiation.

binary down [152] Shape of a parallel binary tile or computation with at least one *up-both* operation.

binary tile [145] A tile in which at most two subcalls may be made.

binary trellis [145] An infinite binary tree structure representing the potential computation of a binary recursion. Each position of the trellis corresponds to the computation performed by one layer of the recursion.

binary up [152] Shape of a parallel binary tile or computation with *up-both* operations.

cable [65] In a sliver decomposition, a collection of wires that connects slivers.

call boundary [130] An abstract barrier that separates the computation of a procedure body from the computation of its arguments and the computation that uses its results. The call boundary is defined by two events: call initiation begins the computation of the body, and call return ends the computation of the body. In tile diagrams, a call boundary is represented as the top line of a tile.

call initiation [131] The event marking the start of a tile computation. All arguments must be available before this event.

call return [131] The event marking the end of a tile computation. All result values must be available before this events.

cell [298] A one-slot mutable data structure supported by OPERA.

channel approach [82] An approach to the signal processing style of programming in which the signal processing operators are represented as processes that transmit data over communication channels.

compaction approach [208] An approach to the filtering of aggregate data in which the output of the filter contains only the elements that pass the filter. Used in contrast with **gap approach**.

computational shapes [129] Computational patterns that characterize how processes unfold over time. See also **shape**.

consumpts [174] Horizontal inputs consumed by a subtile (in contrast with **arguments**, which are vertical inputs). In subtile diagrams, consumpts appear as inputs terminating on the left-hand edge of the subtile. In the context of slivers, consumpts refer to the input slags of a sliver.

conditionally tail-recursive [137] Describes a tile that contains both trivial and non-trivial paths through an *up* shard.

control arm [133] A section of a shard that contains a conditional that must be performed before some subcall.

deadlock [142] A state in which a computation can make no progress even though it has not terminated with a result.

down [132] An orientation for a linear tile operation indicating that the operation must be performed before subcall initiation. Also used to describe the shape of a sliver consisting of only down operations. *Down-both*, *down-left*, and *down-right* tree operations are loosely classified as *down* operations as well.

down barrier [166] A point defined by call initiation that separates the computation of the arguments to a procedure call from the computation of the body of the call.

down-both [148] An orientation for a binary tile operator indicating that the operator must be performed before both the left and right subcall initiations.

down-left [148] An orientation for a binary tile operator indicating that the operator must be performed before the left subcall initiation but is unordered with respect the right subcall.

down-right [148] An orientation for a binary tile operator indicating that the operator must be performed before the right subcall initiation but is unordered with respect the left subcall.

down **phase** [139] The phase of a linear iteration or recursion in which all *down* operations are performed.

DYNAMATOR [400] A graphical program animator for the EDGAR graph reduction model.

eagon [319] An OPERA object embodying an eager evaluation strategy. Equivalent to the **future** objects of other languages.

eager evaluation [158] A non-strict argument evaluation strategy for a procedure application in which arguments are evaluated in parallel with the evaluation of the procedure body.

EDGAR [347] An *E*xplicit *D*emand *G*r*A*ph *R*eduction model. EDGAR differs from most other graph reduction models in its explicit representation of the demand for the evaluation of a subgraph.

element wire [176] In a sliver decomposition, a wire that transmits the element associated with a particular layer of a recursive process.

excludon [314] A mutual exclusion object supported by OPERA.

fan-in [26] A situation in which a device has more than one input.

fan-out [26] A situation in which an output of some device is shared by more than one other device.

filton [336] A data structure representing a filtered element of a synchronized lazy list.

filtered slag [335] A synchronized lazy aggregate whose elements may be gaps.

future [298] Synonym of **eagon**.

gap [177] A distinguished element in an aggregate data structure that marks a position where a value has been filtered out. Written as **#g**.

gap approach [208] An approach to the filtering of aggregate data in which the output of the filter contains explicit gap entities for elements that did not pass the filter. Used in contrast with **compaction approach**.

in [152] Shape of a sequential tree tile having at least one *between* operation and no *up* operations. Also applies to tree slivers and computations whose operations satisfy these criteria.

in-order [269] A tree processing strategy in which the node of a tree is processed after some of its children but before other of its children.

interface tiles [143] Unreplicated tiles that perform initialization and finalization for a replicated tile computation.

intermediates [133] Values that do not cross the call boundary or subcall boundaries of a procedure. In tile diagrams, intermediates are produced by one shard and consumed by another shard within a tile.

iteration [51] Synonym for **linear iteration**.

layer decomposition [65] "Horizontal" decomposition of a recursive monolithic computation that collects together all operations performed in one layer of the recursion.

lazy data [98] Data structures whose components are not computed until their values are required.

lazy evaluation [158] A non-strict argument evaluation strategy for a procedure application in which the evaluation of arguments is suspended until their values are required.

lazon [315] An OPERA object embodying a lazy evaluation strategy. Equivalent to an automatically forced Scheme **delay** object.

left-to-right tile [150] Any tree tile in which dataflow constrains subcalls to be visited in a left-to-right order.

linear computation [51] A process whose dynamic calls form a linear sequence.

linear *down* shape [140] Shape of a linear computation that has no *up* phase. Iterative computations have a linear *down* shape.

linear iteration [51] A linear computation exhibiting constant control space.

linear recursion [51] A linear computation requiring an implicit control stack.

linear tile [130] Tile in which at most one subcall may be made.

linear *up* shape [140] Shape of a linear computation that has an *up* phase.

linear trellis [130] Infinite linear structure representing the potential computation of a linear iteration or recursion. Each position of the trellis corresponds to the computation performed by one layer of the iteration or recursion.

lock step component [171] A collection of slivers that engages in lock step processing to simulate the behavior of a monolithic recursive procedure. Any two slivers communicating via a synchronized lazy aggregate are in the same lock step component.

monolithic [52] Describes a program organization in which programming idioms are manually interwoven in a way that obscures the conceptual structure of the program.

modular [52] Describes a program organization that effectively encapsulates programming idioms into units that can be combined in mix-and-match ways.

multi-threaded [163] Describes a computation with a multiple loci of control.

non-strict evaluation [158] Any argument evaluation strategy in which the evaluation of arguments is not required to precede the evaluation of the body of the applied procedure. Laziness and eagerness are examples of non-strictness.

non-tail call [166] A procedure call that requires the pushing of control stack.

non-waiting pointer [302] A reference to a synchron that blocks a rendezvous from occuring at that synchron.

OPERA [295] A dialect of Scheme supporting concurrency (concurrent evaluation of procedure call subexpresions), synchronization (synchrons and excludons), and non-strictness (lazons and eagons).

operational faithfulness [29] The principle that a lock step component of slivers should simulate the operational behavior generated by a monolithic recursive procedure.

operation order [29] A partial order describing the relative times at which operations may be performed.

orientation [132] A property of a tile operator that specifies when it is performed relative to the subcalls of the tile.

`par` [101] An eager evaluation mechanism introduced by Hughes. Similar to a **future**.

parallel down [164] Shape of a parallel binary tile or computation with no *up-both* operations.

parallel tile [150] Any tree tile in which there is no dataflow between subcalls.

parallel up [164] Shape of a parallel binary tile or computation with at least one *up-both* operation.

post [152] Shape of a sequential tree tile having at least one *up* operation. Also applies to a tree computation performing only these operations.

post-order [269] A tree processing strategy in which a node is processed after all of its children are processed.

pre [152] Shape of a sequential tree tile having only *down* operations. Also applies to a tree computation performing only these operations.

pre-order [269] A tree processing strategy in which a node is processed before any of its children are processed.

presence wire [176] In a sliver decomposition, a wire that transmits a flag indicating whether the current element is present. A presence wire carrying false indicates a gap.

products [174] Horizontal outputs produced by a subtile (in contrast with **results**, which are vertical outputs). In subtile diagrams, products appear as outputs originating on the right-hand edge of the subtile. In the context of slivers, refers to the output slags of a sliver.

recursion [450] Any recurisvely-structured computation requiring non-constant control space. See also **recursion**.

reify [212] In the context of filtering, an operator that turns a gap into a non-gap value.

rendezvous [197] The point in time at which all processes that *could* be waiting on a given synchron *are* waiting on that synchron.

results [131] The values produced by a procedure application. In tile diagrams, they appear as outputs originating at the line representing the call boundary.

reusability [29] The principle that a module (e.g., a sliver) should have a standard interface so that it can be reused in mix-and-match ways.

right-to-left tile [150] Any tree tile in which dataflow constrains subcalls to be visited in a right-to-left order.

sequential tile [150] Any tree tile in which dataflow constrains subcalls to be visited in some sequential order.

series [104] A synchronized sequence structure defined by Waters. It is the basis for a strategy that transforms modular sequence programs into efficient loops.

shape [138] A property that summarizes the operational nature of a computation. When applied to a tile or subtile, shape is a static property determined by the orientations of operators. When applied to the computation generated by a tile or subtile, shape is a dynamic property determined by the orienations of operators that are dynamically performed.

shard [132] The collection of operations within a tile or subtile that share the same orientation.

signal processing style [81] A style of organizing programs as signal processing block diagrams in which information flows through devices that generate, map, filter, and accumulate data. Aggregate data approaches and channel approaches are two common examples of the signal processing style. SPS is an abbreviation for this style.

single-threaded [163] Describes a computation with a single locus of control.

skeletal node [192] One of the nodes that forms the backbone of an aggregate data structure.

slag [190] Abbreviated form of *S*ynchronized *L*azy *AG*gregate.

sliver [65] Fragment of a recursive computation that captures a programming idiom. When used in the conjunction with synchronized lazy aggregates, denotes a procedure that manipulates such aggregates.

sliver decomposition [65] "Vertical" decomposition of a recursive monolithic computation that distributes the recursive structure over each of the components.

space profile [407] A function that describes the space required by a process as a function of execution time.

SPS [81] Acronym for *S*ignal *P*rocessing *S*tyle.

stream [99] An incrementally computed sequence. In this document, I use the term exclusively to mean Scheme's lazy lists.

strict data [409] Data structures that require their components to be fully computed values.

strict evaluation [131] Any argument evaluation strategy in which the evaluation of arguments is required to precede the evaluation of the body of the applied procedure.

subarguments [131] The values consumed by the subcall boundary of a tile.

subcall [130] Any procedure application that appears within the body of a given procedure.

subcall boundary [130] The call boundary associated with a subcall. In tile diagrams, a subcall boundary is represented as a line at the bottom of the tile. A single tile may have several subcall boundaries.

subcall initiation [131] The event that marks the beginning of a subcall's computation. All subarguments must be available before this time.

subcall return [131] The event that marks the end of a subcall's computation. All subresults are available after this time.

subresults [131] The values produced by the subcall boundary of a tile.

subtile [173] A tile fragment that specifies the computation performed by one layer of a sliver.

SYNAPSE [221] A language for manipulating synchronized lazy lists and trees. SYNAPSE is a suite of slag-manipulation procedures embedded in OPERA. The name stands for *SYN*chronized *A*ggregate *P*rogramming *S*tructures *E*nvironment.

`synch` [101] Hughes's construct for synchronizing two demands for a value.

synchron [302] An OPERA object permitting barrier synchronization of a dynamically determined number of processes. Synchrons are at the heart of the lock step processing model embodied by slivers and synchronized lazy aggregates.

synchronized lazy aggregate [190] A dynamically unfolding aggregate data structure carrying synchronization tokens that enable lock step processing among the producers and consumers of the aggregate.

synchronized lazy list [191] A synchronized lazy aggregate that denotes a sequence of elements.

synchronized lazy tree [191] A synchronized lazy aggregate that denotes a tree of elements.

syndrite [191] A synchronized lazy tree. The name stands for *SYN*chronized den*DRITE*.

synquence [191] A synchronized lazy list. The name stands for *SYN*chronized se*QUENCE*.

synter [192] A structure annotated with two synchrons that connects the skeletal nodes of a synchronized lazy aggregate. The name stands for *SYN*chronized poin*TER*.

tail call [169] A procedure call that does not require the pushing of control stack.

tail-recursive [131] Describes the tail call of a recursive procedure. When applied to tile operations or slivers, the term is synonymous with "*down* shaped".

termination wire [176] In a sliver decomposition, a wire that transmits a flag indicating whether there is any more data.

tile [130] A specification of the computation performed by one layer of an iteration or recursion. A tile is depicted as a box whose contents specify the potential operations performed by the layer.

tree shaped computation [56] A process whose dynamic calls form a tree.

unfiltered slag [330] A synchronized lazy aggregate whose elements may be gaps.

unitilable [138] Describes a recursive computation in which every layer is generated by the same tile.

unreify [212] In the context of filtering, an operator that turns a non-gap value into a gap.

up [132] An orientation for a linear tile operation indicating that the operation must be performed after subcall return. Also used to describe the shape of a linear sliver that has at least one up operation. *Up-both*, *up-left*, and *up-right* tree operations are loosely classified as *up* operations as well.

up barrier [166] A point defined by call return that separates the computation of the results of a procedure call from the computation that uses these results.

up **phase** [139] The phase of a linear recursion in which all *up* operations are performed.

up-both [148] An orientation for a binary tile operator indicating that the operator must be performed after both the left and right subcall returns.

up-left [148] An orientation for a binary tile operator indicating that the operator must be performed after the left subcall return but is unordered with respect the right subcall.

up-right [148] An orientation for a binary tile operator indicating that the operator must be performed after the right subcall return but is unordered with respect the left subcall.

waiting pointer [302] A reference to a synchron that indicates that a process is ready to engage in a rendezvous at that synchron.