Efficient computation of embodied energy from a dependency tree

David M. Delaney
142 Waverley Street,
Ottawa, Canada, K2P 0V4
ddelaney@sympatico.ca

v1.5 January 22, 2002

Abstract: Presents a procedure to compute the energy embodied in an object as specified in its energy dependency tree. The procedure uses estimates of the embodied energy of constituent and contributing objects to guide its decisions about which parts of the infinite tree to visit. The costs of objects and services may be used to generate suitable estimates. The accuracy of the procedure is insensitive to the accuracy of the estimates. The procedure visits only the number of nodes of the tree required to achieve the desired accuracy.

Keywords: embodied energy, net energy, invested energy, energy returned, EROEI, EROI, energy tree, energy dependency tree, energy dependency graph.

In the study of energy efficiency and net energy, we must determine the energy *embodied* in manufactured objects. We embody energy when we create an object if we could dissipate that energy in other pursuits were we not to create the object. For example, the electrical energy that powers the tools that work on the object could used for some other purpose, so forms a part of the embodied energy of the object. Less directly, the fraction of the life of a tool devoted to the manufacture of an object could have been devoted to some other purpose. The same fraction of the embodied energy of the tool must, therefore, be considered to be embodied in the object manufactured by the tool. A tiny share of the energy embodied in the factory building was dissipated with the purpose of creating each of the objects made in the factory. This tiny share must also be regarded as part of the embodied energy of each manufactured object. The process of attributing the sources of embodied energy goes on like this indefinitely, encompassing the energy, tools, and factories used to create the tools in the factory in which the object was made, and so on.

To begin accounting for all of these energy contributions to the embodied energy of a manufactured object, write the name of the object on the top line of a very long, very wide, piece of lined paper. Then write a list on the second line, arranged so that the center of the list, is below the object above. The entries of the list include the name of the components that were assembled into the manufactured object, the direct energy inputs required to modify the components and assemble the them into the object, the indirect energies (i.e. names representing them) required to deliver each of the direct energies, the tooling operations and machines required, the human labor required, and the name of the factory, Space these written entries very widely on the second line.

If we know the embodied energies of each of the objects listed on the second line of the paper, and the fraction of its embodied energy that each such object contributes to the object on the first line, we can add up the contributions to get the desired answer. The objects specified by list entries on the second line include, for example, direct energy inputs, components of the object on the first line, tools, factories, and facilities of other kinds. Often, except for the direct energy inputs, we don't know the magnitude of the embodied energies of objects listed on the second line. To compute them we have to repeat the process for every object on the second line, writing a number of lists of objects on the third line, each such list specifying objects contributing embodied energies to one of the objects of the second line. To be sure of having accounted for all energy inputs to the object on the first line, we may have to write many thousands of lines of lists, and write many millions of entries in those lists. In fact, since objects may depend mutually on each other through circular chains of dependence, the piece of paper must, in principle, be infinitely large.

It seems obvious that only a finite and small number of lines near the top of the paper contribute significantly to the energy embodied in the object named at the top of the paper. In other words, if we sum the ultimate energy contributions of each line to the embodied energy of the object at the top of the page, and write these sums in line order, we have written a series whose sum converges rapidly. Moreover, errors in the specification of embodied energies of entries low on the page contribute much less error to the embodied energy at the top of the page than errors in entries high on the page. It is difficult to make direct use of this intuitive appreciation of convergence, but it does give us confidence that a rapidly converging procedure for calculating the sum of the series must be possible.

Let's call the entries written on the lines *nodes*. The *parent* of a node *N* is the node on the line above the line on which *N* occurs and to which *N* contributes embodied energy. If you draw a line between each node and its parent, the resulting drawing looks like an inverted tree, so we call it a tree. The node at the top of the paper is the *root* of this tree. We will accept that this kind of tree has its root at the top of the tree, and its leaves at the bottom. A node is a child of its parent. Two nodes are siblings if they have the same parent. A node *A* is an ancestor of another node *N* if *A* is the parent of *N*, or if *A* is the parent of an ancestor of *N*. Node *D* is a descendant of node *A* if *A* is an ancestor of *D*. The sub-tree based at a node *N* is the set of nodes consisting of the node *N* itself (the *root of the sub-tree*) and all nodes of which *N* is an ancestor.

We use just two types of nodes to represent tree data: direct energy input nodes, and facility nodes. A direct energy input node has no children. It specifies the magnitude of a direct energy input to its parent. A facility node represents an object or process--a factory, tool, job, labor input, or indirect energy input. A facility node always has children. Facility nodes representing indirect energy inputs deserve special mention. A direct energy input always requires additional indirect energy inputs to make the direct energy available. Electricity has to be generated and transmitted to the point of use. Oil has to be extracted and refined and carried to the point of use. So we must have a sibling node of the direct energy node, an indirect energy input node, which describes itself as reporting an indirect energy input to the parent, and which has children whose energy contributions must be summed to find the embodied energy of the indirect energy node. Those children and their sub-trees will include the equipment and energy necessary to extract, transform, and deliver the direct energy of the sibling direct energy node.

What ultimate contribution to the embodied energy of the object at the top of the tree is made by the embodied energy of a node deep in the tree?

Consider a factory object. Only a tiny fraction of the energy embodied in the factory may be attributed to the embodied energy of an individual object made in the factory. Similarly, only a small fraction of the energy embodied in a tool may be attributed to an object on which the tool has worked. To represent the consequences of this shared use of an object by other objects, each node includes a data item called the contribution fraction, *cf*, which indicates what fraction of the energy embodied by a node is contributed to its parent. The *cf* of the root node is 1. The *cf* of a direct or indirect energy node is always 1. The *cf* for a factory entry will indicate what fraction of the embodied energy of the factory is contributed to the parent of the factory entry. (The parent is a node representing an object manufactured in the factory).

The *ultimate contribution factor*, *ucf*, of a node is the fraction of the energy represented by a node (or embodied in an object represented by the node) that is contributed to the embodied energy of the object represented by the root of the tree. The *ultimate contribution* of a node is the product of the *ucf* of the node and the energy embodied in the object represented by the node. The *ucf* of the root of the tree is 1. The *ucf* of any other node is its contribution fraction, *cf*, multiplied by the *ucf* of its parent. We can see that the *ucf* values get very small very fast as we descend the tree. Every object the use of which is shared by multiple objects of which the node representing its parent is one has a contribution factor with respect to its parent that is less than one--often much less than one. It is the rapid decrease of *ucf* values as we descend the tree that gives us confidence in the existence of a converging procedure for calculating embodied energy.

A moment's reflection will reveal that the embodied energy of the object represented at the root of the tree is the sum of the ultimate contributions of all the direct energy nodes of the tree--an infinite sum. As a result of the rapid decrease of the *ucf* values as we descend the tree, it is clear that the ultimate contributions of only a relatively small number of direct energy nodes need to be accumulated in order that their sum should approximate the sum of the ultimate contributions of all direct energy nodes to any desired degree of accuracy. But which direct energy nodes? How do we find them efficiently?

Suppose we had a way to locate and accumulate larger ultimate contributions before smaller ones, and also had a way to estimate the difference between the accumulated sum and the desired result to within a bounded error. We could stop accumulating and declare the result when the estimate of the difference indicated that the real difference was smaller than the desired accuracy.

Such a procedure requires a way of estimating the energy embodied in objects and services. Such estimates would not have to be accurate, but would have to have bounded error. One suitable way of estimating such energies is based on the cost of the object or service, the GDP of the economy in which the object or service was produced, and the total energy, E, consumed by that economy:

energy embodied or consumed = cost x E/GDP

Any estimating method with bounded error, or even several different methods could be used in one tree.

The structure of a node of the tree must be expanded to include a data element, the *estimated nodal embodied energy*, or *enee*, to represent the estimate of the energy embodied in the object represented by the node. The value of *enee* for a direct energy node is equal to its embodied energy . We define the *estimated ultimate contribution* of the node as the product of the *enee* and the *ucf* of the node.

The accuracy of an evaluation procedure conforming to this sketch is not sensitive to the accuracy of the method(s) of estimating the energy embodied in objects or services. (For simplicity this discussion assumes that the energy specification of a direct energy node is accurate. Recall that the *enee* of a direct energy node is equal to the direct energy contribution of the node.) Estimates that are too large increase the number of nodes that must be inspected, and therefore the time required to compute the desired result, but have no effect on the accuracy of the result. Estimates that are much too small could, in principle, impair the accuracy of the result of the computation. To see this, recall that large ultimate contributions are accumulated before small ones. If the estimate of the ultimate contribution of a node were much too small, the procedure may have stopped before accumulating the significant ultimate contribution of the underestimated node. Practically, however, an estimate would have to be very much too small to produce a significant error in the procedure. First, it is fairly easy to ensure that estimates are not too small. Second, since larger ultimate contributions are collected first, a low estimate of a larger ultimate contribution, but not wildly low, will still be larger than estimates of the tiny contribution that will be the last one considered, delaying the accumulation of the large contribution, but not affecting the accuracy of the result. Third, details of procedure can be introduced to check for wild inconsistencies between the estimated contributions of the children and the estimated embodied energy of the parent. Finally, the contribution of errors introduced by low, but not wildly low, estimates of embodied energies may be reduced to arbitrarily low values by having the stopping criterion demand a sufficiently small estimated difference between the accumulated sum and the desired result.

To this point we have assumed that we can efficiently accumulate larger ultimate contributions before smaller ones. A procedure for doing so is described below . First we define variables, functions, predicates, and constants.

The variable *ee* (embodied energy) accumulates the sum of the ultimate contributions of processed direct energy nodes to the embodied energy of the object represented by the root of the tree.

The variable *nun* (next unprocessed node) designates the next node to be processed by the procedure.

The variable *ucpn* (unprocessed children of processed nodes) represents a set containing designations of all unprocessed children of processed nodes.

The function *estimated_error* computes and returns as its value the sum of the estimated ultimate contributions of the unprocessed children of processed nodes (*ucpn*). Its value is zero if *ucpn* is empty.

The constant *allowed_error_f raction* represents the fraction by which the desired embodied energy may be in error.

The predicate *direct_energy*(n) is true if and only node n is a direct energy node.

The predicate *children*(n) is true only if and only if node n has children.

The brackets /* and */ enclose comments that are not part of the procedure.

```
Procedure EBE;    /* embodied energy evaluator  */

begin

     ee := 0;
     put a designation of the root node of the tree as the only entry of ucpn ;

     repeat

          nun := a designation of the node in ucpn that has the largest
                  estimated ultimate contribution of all nodes in ucpn ;
                     /* nun is "unprocessed" */
          if direct_energy(nun)  then
               ee := ee + ((ucf  of node nun)* (direct energy specified by node nun));
          else
               if  not  children( nun) then      /* facility node must have children */
                   obtain the children of nun from a competent source;
               fi;
            place a designation of each child of  nun in ucpn ;
          fi;
          remove nun from ucpn;
          /* nun is "processed" */

      until allowed_error _fraction * ee  > estimated_error ;

    write(  "The embodied energy of  the root of the tree is ",  ee  );

end EBE;
```

At the end of each iteration of its "while" loop EBE tests for termination. Note that estimated_error returns zero when *ucpn* is empty. The loop terminates when the allowed error in the accumulated embodied energy is less than the estimated error. The estimated error is the total ultimate contribution of all unprocessed children of processed nodes (zero if *ucpn* is empty). At the beginning of the body of the loop *ucpn* is non-empty. EBE first chooses the node of *ucpn* (unprocessed children of processed nodes) that has the largest estimated ultimate contribution. If the chosen node is a direct energy node, its ultimate contribution is added to *ee*, otherwise EBE asks for the children of the chosen node and includes them inthe set of unprocessed children of processed nodes. As the final step of each iteration before the test for termination, EBE removes the newly processed node *nun* from the set of unprocessed children of processed nodes (from *ucpn*.)

The following statements are true at the beginning and end of each loop iteration: The parent of every node designated in *ucpn* is a processed node. No node designated in *ucpn* is an ancestor of any other node in *ucpn*. The children of a facility node in *ucpn* are unprocessed. The only unprocessed nodes that have processed parents are nodes designated in *ucpn*. Every unprocessed node not in *ucpn* has exactly one ancestor facility node in *ucpn.*

These statements imply that the nodes designated in *ucpn* form a thin ragged fringe that separates the tree into three sets of nodes: an upper portion of the tree all of whose nodes have been processed, the fringe itself, and a lower portion of the tree none of whose nodes have been processed. None of the fringe nodes have been processed. The fringe is everywhere one node deep except where there are gaps in the fringe where processed direct energy nodes dangle. (Direct energy nodes have no children.) The fringe (*ucpn*) dips down where there are nodes that have larger ultimate contributions than other nodes at the same level in the tree.

Since any unprocessed node not in *ucpn* has exactly one ancestor facility node in *ucpn*, all ultimate contributions of such nodes are included in estimated ultimate contributions of the facility nodes in *ucpn*. Since no node in *ucpn* has an ancestor in *ucpn*, the sum of the estimated ultimate contributions of the nodes in *ucpn* (the value of *estimated_error*) contains no double count of estimated ultimate contributions. Recall that the embodied energy of the root of the tree is the sum of the ultimate contributions of the infinite number of direct energy input nodes of the tree. We can conclude immediately that *estimated_error* would be equal to the difference between *ee* and the embodied energy of the root of the tree if the estimate of embodied energy specified by each node in *ucpn* were perfectly accurate (if the *enee* of the node were equal to its embodied energy). We can further conclude that if every estimate of embodied energy in the tree is either accurate or too large, then the error in taking *ee* as the embodied energy of the object at the root of the tree is less than or equal to the value computed by *estimated_error.*

Experienced computer programmers may note that data structures designed for efficient representation of *ucpn* would include at least one priority queue.

We need only a finite piece of paper for the tree used by procedure EBE, because it asks for the input of only a small number of nodes. By locating and summing large contributions to the desired embodied energy before smaller ones, EBE minimizes the number of nodes inspected.

Although the energy embodied in a node is, by definition, equal to the sum of its children's contributions, it is *not* necessary for accurate operation of EBE that the *estimate* of the energy embodied in the node should equal the sum of its children's *estimated* contributions, provided that there are no wild inconsistencies. EBE can be modified to check and warn of wild inconsistencies.

The properties of EBE have important implications for the preparation of a library of embodied energy specifications. Such a library speeds up the operation of EBE, and, more importantly, enormously simplifies the preparation of data for the evaluation by EBE of a new object. For the first (or any) object to be processed for the library, EBE will ask for the nodes it needs. It won't ask for nodes that contribute only insignificantly relative to the desired accuracy of the result. If a subsequent evaluation by EBE of a new object requires an object from the library as a node, the node entered to represent the object may be entered as a special kind of direct energy node having an energy equal to the embodied energy of the library object, and a contribution factor appropriate to its relationship with its parent. Its absence of descendants will greatly speed up its evaluation. When EBE asks for nodes representing nodes not in the library, estimated embodied energies may be entered to get an indication of where the low points of EBE's descent into the tree will occur. Such indications can be of assistance in organizing the work of gathering data.

End of document.