Optimizing Exergy-Services Supply Networks for Sustainability

Robbie Morrison

A thesis submitted for the degree of
Masters of Science
at the University of Otago, Dunedin,
New Zealand

31 July 2000

Abstract[1]

Our current energy systems, particularly those reliant on fossil fuels, are — in terms of resource use, climate change, and local impacts — highly unsustainable.

This thesis presents a generic energy system model that can be used to identify changes in system architecture, replacement technologies, and demand patterns which reduce some chosen suite of sustainability costs — for instance, depletable fuel use, CO_2 output, and local air pollution — whilst maintaining energy-service levels. The model also tallies monetary cost so that beneficial changes can be traded against financial penalties, should these arise.

The model was developed at the University of Würzburg, Germany, and programmed as the UNIX-based application *deeco*: dynamic energy, emissions, and cost optimisation. *deeco* provides a numerical modelling environment for undertaking energy system optimisation of the type just described and includes a library of common plant types.

Mathematically, the model classifies as a dynamical flow network optimisation problem. The flow network itself is best described in terms of exergy, although the network currency used by *deeco* is energy. Exergy-service demand drives the problem.

The model is constructed as follows. An energy system of interest is abstracted as a collection of interconnected discrete plant. The plant are treated as dynamic objects, with their intertemporal energetic input/output behaviour, capacity limits, and fixed and flow-dependent costs encoded as functions or inequalities as appropriate. Abutting plant are interfaced using logical exergy connections to form a graph-theoretic flow network — the physical structure. Time-series data-sets representing exergy-service demand by location and the prevailing ambient and institutional conditions — the informational structure — complete the problem specification.

[1] The author can be contacted at: `robbie@physics.otago.ac.nz`.

After selection of a flow-linear cost goal to proxy for sustainability, the model steps through a sequence of time-intervals (8760 hourly intervals by default) and, assuming redundancy, optimises the flow routing — that is, plant usage — for each interval. Specialist algorithms resolve heat-exchange conditions and store and export surplus exergy between intervals — given certain restrictions on inter-plant influence and abutting network behaviour for reasons of tractability. The storage policy implemented is non-anticipatory, but dynamic programming techniques could facilitate intertemporal optimisation. The key modelling requirements are that the marginal plant efficiencies be independent of duty, or approximated as stepwise-decreasing, and that the selected optimisation cost be linear on flow, or approximated as piecewise-increasing. The marginal plant efficiencies may be arbitrarily dependent on prior state and on ambient conditions. Upon completion, the model reports plant usage and aggregated cost statistics for subsequent interpretation.

As well as providing quantitative decision support, the model also portrays energy policy concepts — such as efficiency, renewable energy, demand management, use of storage, waste recovery, and merit-order dispatch — as interdependent components of a more general dynamical flow network optimisation problem.

The thesis also extends the concepts of exergy quality and intra-plant quality matching, and advocates the use of quality mismatch when searching for potential infrastructural improvements.

The thesis concludes with a review of New Zealand energy sector policy problems that may benefit from quantitative modelling using *deeco*. □

Acknowledgments

My sincere thanks to my initial supervisor, Professor Gerry Carrington, Physics Department, to my co-supervisor, Dr. Chris Handley, Computer Science Department, and to the Head of the Physics Department, Professor Ian Hodgkinson.

This work also drew upon ideas and assistance from the following people. Dr. Thomas Bruckner[1] generously made available the source code for *deeco0.5*, provided test data, and answered my many questions. Dr. Helmuth Groscurth[2] spent several weeks in Dunedin helping with the development of *deeco0.6* and discussing modelling options for New Zealand. Dr. Dietmar Lindenberger, Dr. Zhifa Sun, Eli Yasni, and Molly Melhuish offered valuable academic insight and criticism. Dr. Richard O'Keefe, Conal Tuohy, Scott Dunavan, Malcolm Fraser, and Brent Russell provided software design, programming, and systems support. Jan Diettrich translated key material from German. Ilse Seilis corrected and improved this manuscript. The 1998 EMAN 402—Advanced Energy Analysis class provided constructive feedback for some of the material on exergy. I would like to express my gratitude to all those who contributed. □

[1] Energy Conversion and Protection of the Environment Section, Institute for Energy Engineering, Technical University of Berlin (Technische Universität Berlin), Berlin, Germany. Formerly, Integrated Systems Analysis Department, Potsdam Institute for Climate Impact Research (Potsdam-Institut für Klimafolgenforschung e.V.), Potsdam, Germany, and prior, The Institute for Theoretical Physics, University of Würzburg (Universität Würzburg), Würzburg, Germany.

[2] Special Projects Manager, Hamburg Electricity Works (Hamburgische Electricitäts-Werke AG), Hamburg, Germany. Formerly, Department of Environmental and Resource Economics, Centre for European Economic Research (Zentrum für Europäische Wirtschaftsforschung GmbH), Mannheim, Germany, and prior, The Institute for Theoretical Physics, University of Würzburg (Universität Würzburg), Würzburg, Germany.

Contents

List of figures

Abbreviations

AC	:	alternating current
AEPSOM	:	Australian energy planning system optimization model
aka	:	also known as
ANSI	:	American National Standards Institute
API	:	application programming interface
ARM	:	Annotated C++ reference manual, a *de facto* standard
ASCII	:	American standard code for information interchange
AT&T	:	American Telephone and Telegraph Company
AUP	:	acceptable use policy
BAU	:	business-as-usual
BSD	:	Berkeley UNIX
CBA	:	cost-benefit analysis
CD	:	compact disk
CFD	:	computational fluid dynamics
CGE	:	computational general equilibrium
CGI	:	common gateway interface protocol
CHP	:	combined heat and power
CLI	:	command line interface
COP	:	Conference of the Parties, conducted under the FCCC
COP	:	coefficient of performance
CO_2-e	:	CO_2-equivalent (see p 174)
CPU	:	central processing unit
DC	:	direct current
DCF	:	discounted cash flow analysis
DCM	:	data-channel multi-digraph
deeco	:	dynamic energy, emissions, and cost optimisation
DH	:	district heating
DOE	:	US Department of Energy
DOS	:	indicates MS-DOS PC operating system
DP	:	dynamic programming

DR	: distributed resources
DSM	: demand-side management
ecco	: energy, cost, and carbon dioxide optimization
EDG	: Edison Design Group
EERA	: energy efficiency resource assessment
EID	: exergy interconnections digraph
EFOM	: energy flow optimisation model
eg	: for example
EIA	: Energy Information Administration, DOE
EMRG	: Energy Modelling Research Group
ERA	: entity–relationship–attribute modelling
ERS	: environmental reference system
ES	: $nergy system
ESSS	: exergy-services supply system
etc	: and so on
EU	: European Union
FAT	: file allocation table
FCCC	: UN Framework Convention on Climate Change
FFE	: far-from-equilibrium (thermodynamics)
GDP	: gross domestic product
GHG	: greenhouse gas
GNU	: acronym for freely-used UNIX-compatible software project
GP	: gravitational potential (exergy)
GUI	: graphical user interface
GWP	: global warming potential
HEN	: heat-exchanger network
HHV	: higher heating value
HP	: Hewlett-Packard
HTML	: hyper-text mark-up language
HTTP	: hyper-text transfer protocol
HX	: heat-exchanger
IDEAS	: integrated dynamic energy analysis simulation
ie	: that is
IEA	: International Energy Agency
IEEE	: Institute of Electrical and Electronic Engineers
iff	: if and only if
IIASA	: International Institute for Applied Systems Analysis
i/o	: input-output
IP	: integer programming
IPCC	: UN International Panel on Climate Change
JI	: joint implementation
LCA	: life-cycle analysis
LEDA	: a library of efficient data types and algorithms

LHV	: lower heating value
LMTD	: log mean temperature difference (equation)
LP	: linear programming
LTE	: local thermodynamic equilibrium
MCFP	: minimum cost flow problem
MESSAGE	: model for energy supply systems analysis and their general environmental impact
MIS	: macroeconomic information system
MODEST	: model for optimization of dynamic energy systems with time-dependent components and boundary conditions
MP	: mathematical programming
MP	: multi-processor
MS	: Microsoft
MW_e	: MW electrical
n/a	: not applicable
NDP	: network design problem
NEMESS	: network model of energy-services supply systems
NEMS	: national energy modelling system
NET	: non-equilibrium thermodynamics
NMVOC	: non-methane volatile organic compounds
NSSF	: non-steady-state steady-flow
NZEM	: New Zealand electricity market
OHP	: overhead projector (slides)
OOP	: object-oriented programming
OR	: operational research
OS	: operating system
OSS	: open-source software, a software development model
pa	: per annum
PC	: IBM-compatible personal computer
PCE	: Parliamentary Commissioner for the Environment
PERT	: project evaluation and report technique
PM_{10}	: particulate matter under $10\,\mu$m in diameter
POSIX	: IEEE portable operating system interface standards
ppb	: parts per billion
ppm	: parts per million
RAM	: random-access memory
RC	: resistor/capacitor (circuit)
RDBMS	: relational database management system
RISC	: reduced instruction set computer
RMA	: Resource Management Act 1991 and ammendments
RMS	: root mean square
SC	: Standard Components
SCO	: Santa Cruz Operation

SDK	: SCO software development kit
SMP	: system marginal price
SOLEG	: solar-supported energy supply for buildings
SPD	: schedule, price, and dispatch
SQL	: structured query language
SRE	: storage replenishment and export algorithm
SSSF	: steady-state steady-flow
STL	: standard template library
STP	: standard temperature and pressure
SVRn	: System V release n, a UNIX lineage
TLFO	: time-local flow optimisation algorithm
TM	: thermomechanical
TRS	: thermodynamic reference system
TSN	: thermal sub-network algorithm
UDK	: SCO UnixWare and OpenServer development kit
UN	: United Nations
URL	: universal resource locator
USL	: UNIX Systems Laboratories
VBA	: Visual Basic for applications
WEMDG	: Wholesale Electricity Market Development Group
WWW	: world-wide web
x86	: indicates Intel-compatible 386DX or higher processor

Chapter 1

Introduction

This chapter outlines the thesis topic, notes key literature, and lists external relationships.

1.1 Overview of the thesis

The energy systems which fuel industrial nations are increasingly being viewed as unsustainable in their present form. This thesis presents a method of modelling these systems — at any required level of jurisdiction — in order to identify opportunities which improve aggregate sustainability whilst maintaining similar levels of energy-service.

Much of this thesis centres on the UNIX-based application program *deeco* : dynamic energy, emissions, and cost optimisation — which provides a supported numerical environment for such modelling.

The overall structure of this document can best be seen by scanning the contents listing. Readers should note that chapters 2 and 3 can be omitted with little loss of continuity. Alternatively, those interested in *deeco* specifically can read chapters 1 and 8 in isolation.

This section undertakes three tasks. First, it introduces the energy network optimisation model central to this thesis. Second, it describes the use of this model as a quantitative decision support tool for improving energy system sustainability. And third, it reviews *deeco*.

Note on terminology

This thesis spans a number of disciplines and, as a result, certain technical terms attract conflicting definitions. To resolve this difficulty, a number of terms have been attributed specific meanings as given in appendix A. Readers seeking precision may wish to review these definitions before proceeding.

Energy-related terms

Five terms related to *energy* warrant explanation at the outset. These terms are given in table 1.1 and are used as shown. This somewhat unorthodox scheme was selected as being the least clumsy way of differentiating between the various usages of the word *energy*.

Terminology		**General symbol**	**SI unit**	**Comment**
Energy	energy	–	J	Arises jointly from primitives in thermodynamics and mechanics as a system property.[1] Energy can be neither created nor destroyed, but it may be converted in form.
ℑnergy	εnergy	L, Λ	J	Potential of a flow or stock resource to produce (low-grade) heat, and dependent on prevailing environmental conditions.
Exergy	exergy	E, Ξ	J	Potential of a flow or stock resource to produce (high-grade) work, and dependent on prevailing environmental conditions.
$nergy	$nergy	Σ [3]	J, $ *etc*	Economic factor of production or consumption and therefore limited to flow commodities. $nergy bundles concepts of exergy content, production convenience, and economic value, and is thus dependent on prevailing environmental, technological, and market contexts. In this thesis, $nergy is quantified thermodynamically using truncated flow exergy, and economically using opportunity cost.[2] In practice, $nergy is usually accounted using truncated flow εnergy.
'Energy'	'energy'	–	–	The word energy set in single quotes indicates its colloquial meaning is intended.

Table 1.1 *Energy-related terminology adopted in this thesis (the table annotation relates to the footnotes below).*

[1] A theoretical primitive is a concept that cannot be reduced further and which has been accepted as the starting point for the establishment of a particular discipline. Within classical mechanics, the concepts of force and the particle are taken as primitives (Slattery 1981). For Gibbsian-based thermodynamics, internal energy U, temperature T, and the simple system may be taken as primitives, although these in turn can be defined using more fundamental concepts from statistical physics. Extending the Gibbsian view yields total energy $\bar{E} = U +$ macroscopic forms of energy, with the latter terms arising from mechanics.

[2] Equating value and price is a cornerstone of neoclassical economics, but nonetheless, value and price may diverge significantly, particularly where externalities exist or some other systematic bias is present. Furthermore, valuing the non-commodity or intangible benefits of network-delivered goods and services remains a research topic.

[3] Σ is at least a 2-element vector, as discussed on p 71.

Of the first four technical variants, energy is the more fundamental concept and, when specifically defined, becomes a state function.[4] Exergy and ϵnergy [5] (see p 58 and p 60, respectively) are derived quantities and their mathematical definitions are cross-referenced in table 5.3 (see p 56). Exergy and ϵnergy arise in flow and nonflow forms, which apply to flow and stock commodities respectively. $nergy (see p 71) is an economic concept and is taken to include information about exergy content, price, and production-related characteristics as appropriate.

Lay usage of the word ′energy′ most closely aligns with the concept of exergy. One contemporary dictionary (Deverson 1997) gives the technical definition of *energy* as the "capacity of matter or radiation to do work" (p 345) and *work* as "the exertion of force overcoming resistance or producing molecular change" (p 1231). Therefore the following advice is offered:

> **Exergy**: readers unfamiliar with the term *exergy* should mentally substitute the word ′energy′ as used in its colloquial sense.

Each of the quantities listed in table 1.1 will give a different account for the same commodity. For example, geothermal steam can be evaluated using flow ϵnergy L, flow exergy E, and $nergy Σ, and each will provide a different assessment of the resource. Ideally, the choice of measure should depend on the purpose of the assessment, but practical considerations may intrude such as the need to align with existing reporting protocols.

The thermodynamic notation used in this work is broadly consistent with Bejan (1997) but with the symbol for total energy represented thus: $\bar{E}$. An engineering sign convention is likewise adopted (see p 19).

Exergy versus ϵnergy

The network model presented in this thesis is framed using exergy to represent the flow and stock commodities of interest. Exergy is chosen over ϵnergy because it more closely aligns with the social concept of value associated with this class of commodity — as evidenced by the lay meaning of ′energy′ (see above).

In addition, two technical reasons indicate the use of exergy over ϵnergy. First, the flow of exergy can be guaranteed to be anti-parallel to the propagation of demand signals within a network — ϵnergy lacks this property because its flow is not necessarily consistently directed with respect to demand (*eg*, heat conduction above and below ambient). Second, the use of exergy removes the need to include zero-exergy

[4] For instance, ′total energy′ is defined in eqn (B.18) (see p 191).

[5] The concept of ϵnergy is novel to this thesis.

ϵnergy (*aka*, anergy) flows within the network representation, thereby reducing its complexity.[6]

Exergy-service and exergy-services supply systems (ESSS)

The fundamental purpose of the systems under consideration is to supply exergy-services in some desired form. Exergy-services are those services that arise, or could have arisen, from the active consumption of exergy. This definition is not particularly specific but examples of exergy-service can be readily listed:

- warmth and refrigeration-services,
- illumination,
- materials extraction and processing,
- movement of goods,
- access to suitable venues to conduct social and commercial interactions.

Certain activities can prove awkward to characterise in terms of exergy-service, in which case their specification may best be represented as outright exergy demand. For example, the electricity used to power data processing equipment is typically represented as [kWh] or [J].

The term ′exergy-services supply system′ or ESSS is taken to describe the sum of all interconnected exergy-services supply networks.[7] An ESSS may include exergy-service supply options that do not necessarily require $nergy— either explicitly priced by trading or implicitly priced through opportunity cost. One such example is the insulating of dwellings so that active heating is displaced. The term ′$nergy system′ or ES is used to describe the system which procures primary $nergy and supplies consumer $nergy as a factor input to firms, a direct private good to households, or an indirect public good to citizens (*eg*, street lighting). Therefore, physically and biophysically, an ES will always be a (null, proper, or equivalent) subset of the ESSS which covers the same problem. Fixing the boundaries of such systems requires modelling judgements made in context of the problem being addressed. An ESSS and an ES both contain physical and institutional components — examples of the latter notion include the statutory rights and responsibilities of participants, and if $nergy is involved, the market forums used to facilitate trading. This work concentrates on the physical and biophysical aspects of ESSS, rather than their economic and institutional interpretation.

[6] Under the model being articulated, the network ′currency′ need only be linear on the flow-dependent cost selected for optimisation. As the intensive state of the network is fully characterised prior to each time-local optimisation exercise, exergy and ϵnergy both comply as potential currencies. *deeco* itself accounts for the commodity stocks and flows using ϵnergy for reasons of convenience, primarily because most plant performance data is published in first law form only. Notwithstanding, the overarching model is best framed in terms of exergy, for the reasons outlined.

[7] The acronym ESSS is also used in the *deeco* literature where it stands for ′energy-services supply system′ and describes *only* that part of the system being subject to optimisation.

Rationale for this thesis

This thesis is premised on the view that present generations are ethically obliged to consider the foreseeable needs of future generations — a condition similar to that written into New Zealand environmental law [8] — therefore, improved exergy-services sustainability (see p 31) is selected as the overarching system goal for such systems.

This thesis argues that the exergy-services sustainability policy debate has failed to keep pace with recent developments, particularly those associated with technical progress and market liberalisation, and that this lack of sophistication can best be redressed by interpreting ESSS as follows:

- using the second law constructs of **exergy** and **exergetic efficiency**, rather than the first law constructs of ϵnergy and ϵnergetic efficiency — primarily because exergy accounting, in contrast to ϵnergy accounting, naturally locates and quantifies wastage along any given exergy-service supply chain, and furthermore provides useful insight into likely remedies,[9]
- as **dynamical systems** — due to the fact that system components (both physical and informational) may interact across discrete time,[10]
- as **flow networks** and optimised as such — because this viewpoint facilitates the overall integration of network architecture, plant performance, and end-user demand.[11]

These three perspectives complement each other. The first is particularly useful when searching for potential infrastructure design improvements, the second when considering issues of storage and load shifting, and the third when considering locational demand management and optimal plant utilisation. The three perspectives, when combined into a single overarching model, give rise to a more lucid view of ESSS than that previously available.

The use of exergy analysis should not, in itself, impose a requirement for more data — given that appropriate thermophysical properties are published for all compounds in use. This is because an exergy balance needs *no more* measured (*ie*, problem-specific) information than does an ϵnergy balance at the same spatial resolution and level of precision (although an ϵnergy balance may be need resolve unmetered losses).

A dynamical system is one where the current state is dependent on its immediate

[8] Refer to s 5(2)(a) of the Resource Management Act 1991, although this statute expressly excludes the issue of mineral depletion.

[9] Exergetic efficiency does not attract a unique definition in the literature — the form used in this thesis is given in eqn (5.15) (see p 64).

[10] In this thesis, a 'dynamical system' is one in which $x_{t+1} = T(x_t)$, with $x \in X$ and $T : X \to X$. Hence, the points x in space X evolve iteratively under the mapping T. T need not be invertible.

[11] The term 'network' is used in its mathematical sense in this thesis (see p 173).

past state. In the case of ESSS, dynamical effects make up only part of the picture, but nevertheless require inclusion.

A flow network representation (the ′flow′ qualifier is often omitted) requires that two conditions be fulfilled: that the physical system can be approximated as interconnected discrete plant, and that the demand requirements and supply offers faced by the infrastructure are known in advance (*ie*, most modelling is based on historical data). In the case of ESSS: demand is the demand for exergy-service, the network commodity is generic exergy in various forms, and all exergy is ultimately sourced from the biophysical environment. There is no mathematical requirement for the network structure to be physically contiguous.

The concept of exergy-services sustainability is interpreted as the need to transform exergy-services supply infrastructures and/or end-user exergy-services demand profiles such that the system produces less irrevocable environmental damage and less resource depletion including that of assimilative capacity.

Multi-criteria optimisation is necessary for problems as complex as exergy-services sustainability. If a single proxy for exergy-services sustainability is required, then arguably net CO_2-e release (see p 174) is the most applicable, assuming that local impacts are held at acceptable levels. Net CO_2-e often correlates well with both resource depletion and — given the current state of knowledge — with global warming costs.

The 1995 UN Intergovernmental Panel on Climate Change (IPCC) predicted that atmospheric CO_2-e concentrations would, as a lower bound, double during the 21st century, primarily as a result of the rate at which fossil fuels are oxidised to provide downstream exergy. Furthermore, the Panel anticipated that consequential climate instability, temperature ramp-rate, and sea-level rise will all occur on scales unprecedented in human history.

The need for a more sophisticated view of ESSS is being driven by a number of parallel developments (the policy-specific items are applicable to New Zealand at least):

- distributed and/or multi-product technologies are becoming viable in a wider range of applications, particularly as technological development and market liberalisation proceeds — conversely, monolithic technologies are becoming increasingly vulnerable to stranding. Reticulation grids are responding to suit — in terms of exergy carrier diversity, interconnection, and overlap, whilst capacity margins are expected to fall and prioritised interruptibility rise,[12]

[12] For example, New Zealand's national grid operator Transpower recognises distributed generation will have major implications for its own operation (Fuge, Little, and Ward 2000).

- end-users are starting to become more time- and place-of-use flexible, a trend which should continue as ′network aware′ and ′grid-operator controllable′ end-use equipment penetrates — in concert, data relating to the state of networks and/or their affiliated markets, is being collected, transmitted, and processed in greater measure,
- environmental costs are increasingly being internalised and/or environmental impact entitlements are becoming more difficult to obtain, particularly where more ′efficient′ alternatives can be shown to exist,[13]
- depletion pricing should begin to feature as key exergy-related resource extraction rates peak and taper and as crown-owned resource rentals rise.[14]

It is therefore becoming less tenable for public and private policy analysts alike to view demand solely in terms of consumer $nergy, to ignore important characteristics imposed by the network structure of the ESSS with which they deal, and to regard exergy accounting as too abstract to be of use.

To date, the New Zealand government has primarily limited itself to promoting localised *ad hoc* technology-based solutions, aimed at improving customer-domain efficiency and the uptake of small-scale renewable sourcing.[15] These policy deficiencies could be rectified to some significant degree by adopting the policy framework and analytical tools presented in this thesis.

Network integration

The research proposal for this thesis began with the sentence: "Analytical approaches are needed which better integrate issues of energy use, associated impact on ecosystem function and adaptation, and long-run supply." The proposal also identified two further concepts as being central: first that, in general, demand is for exergy-service not exergy *per se*, and, second, exergy supply systems naturally represent as interconnected networks.

However, the analytical methods suggested in the proposal were not as general as they might be. A subsequent literature search located the UNIX-based application

[13] Environmental impact entitlements are discussed on p 36. The need for efficiency of use is contained in s 7(b) of the Resource Management Act 1991. Recent case law has interpreted such efficiency in economic terms, but the RMA itself is not specific in this regard.

[14] In New Zealand, hydrocarbon recovery and mineral extraction royalties could well increase. Furthermore, ′hydro′ royalties have been proposed now that major dams (*eg*, the Clyde dam) no longer return their dividends to the state (*National Business Review* 1998). The assimilative capacity of the environment can be interpreted as a limited resource, in which case carbon taxation could be interpreted as a resource rental. Similarly, tradable emission permit schemes would establish emission rate entitlement as scarce.

[15] For example, the New Zealand Parliamentary Commissioner for the Environment recently tabled a report on ′more sustainable energy systems′ (Parliamentary Commissioner for the Environment 2000). This report fails to consider systematic integration as a possible response. See also comments regarding system-wide efficiency assessment contained on p 113.

program, *deeco*, which provides a numerical modelling environment for the network-based cost optimisation of ESSS — the key paper being Groscurth, Bruckner, and Kümmel (1995) which describes the model framework NEMESS used to code *deeco*. The *deeco* methodology represents a substantial advance on the supply chain analysis originally envisaged by the writer.

Groscurth *et al* describe how the built systems which supply exergy-services to households and firms can be viewed as flow networks, and further, given certain technical restrictions, demonstrate how these networks can be programmed using suitable data structures, solved to determine their intensive thermodynamic state, and optimised to reduce some flow-linear sustainability cost such as depletable fuel use or net CO_2-e release.

Exergy[16] *en route* from supply point to end-use process may be variously carried by transport plant and transformed using conversion plant. Plant performance is encoded using input-output equations (*ie*, performance curves) which may be influenced by ambient conditions. Closure requires that the attributes of connected output and input exergy flows (including the appropriate intensive variables) must match. The flow networks themselves are represented using nodes and links.

As previously noted, a flow network specification requires knowledge of commodity requests and supply offers by node, as well as a definition of the physical infrastructure. The end-use processes request exergy in order to fulfil the exergy-service demands they face. Setting aside the issue of dedicated storage and imports from abutting networks, this exergy must ultimately derive from: the extraction of depletable resources, the harvest of renewable sources (*eg*, solar insolation[17]), or unused co-production recovered from plant already invoked (*eg*, waste-heat). Improved plant efficiency can often displace upstream supply requests. Figure 1.1 depicts the situation just described, although the model is not limited to planar networks.[18] However, certain restrictions can apply regarding cyclic structures (see p 70).

Flow-dependent costs representing some adverse consequence can be assigned to each link. Using this information, the network-wide cost creation rate can be calculated for any given network specification and routing pattern.

A network is deemed 'well integrated' in context of its specification (which includes the demand profile) and selected flow-linear cost criteria *if and only if* its routing configuration realises a minimum network-wide cost creation rate. Network opti-

[16] More precisely, flow exergy, but by convention the qualifiers 'flow' and 'nonflow' may be omitted when implied by context.

[17] Specialist terms including 'solar insolation' are contained appendix A.

[18] Planar networks are those which can be drawn without requiring links to cross.

[19] The diagram could be extended to include storage plant. Such plant would toggle from one side of the network to the other depending on whether they were in discharge or replenish mode.

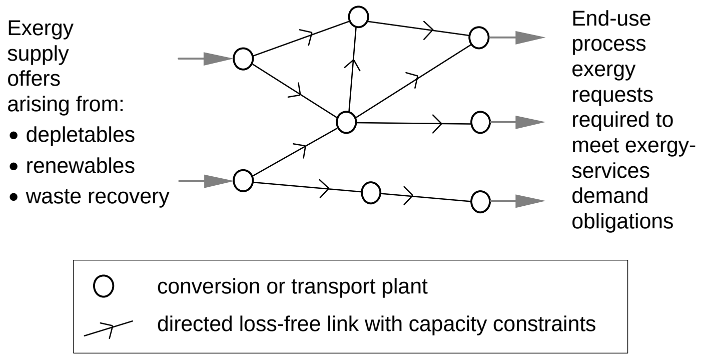


Figure 1.1 *A simplified exergy flow network lacking storage capacity. Although final demand is specified in terms of service, the end-use processes generally require exergy input.*[19]

misation is the process of seeking such a configuration and is only applicable to networks which exhibit redundancy — a situation whereby more than one routing pattern exists. If the network problem is linear, then the underlying equations will be underdetermined,[20] and robust operational research (OR) algorithms, first introduced in the late 1950s, can be used to undertake the task of locating a least-cost flow regime numerically (see p 87).[21]

The method described naturally suits ESSS sustainability-related costs, such as net CO$_2$-e emissions, depletable fuel use, and sulphur release, as these tend to be flow-dependent (*ie*, variable). In contrast, ESSS monetary costs tend to be substantially flow-independent (*ie*, fixed), particularly within a decision context — thereby making them less tractable to numerical optimisation. Nonetheless, the method can be used heuristically — by revising and rerunning the model — to guide the search for lower total cost solutions.

Plant characterisation

The individual plant are characterised using (sets of) process input-output equations — which capture the ϵnergetic efficiency of the plant concerned over its operating range. In order that the flow network optimisation procedure is tractable to linear programming (LP) techniques, these input-output performance must be *independent*

[20] A set of linear equations is ′underdetermined′ if the number of unknowns exceeds the number of independent equations. Furthermore, the presence of lower and upper flow bounds on the links does not render the problem nonlinear, although the behaviour of the network may become less intuitive.

[21] Readers may notice a similarity between the method described and the nodal pricing of wholesale electricity. In the case of nodal pricing, the optimisation goal is to minimise the aggregate flow-weighted demander price, assuming that transmission losses are quadratic and that available supply is best allocated by virtue of localised market clearance at each node. Unlike the *deeco* model, nodal pricing can cope with restricted primary supply and/or constrained transmission.

of duty, but may be *arbitrarily dependent* on external conditions.

The restriction just outlined may be circumvented in *some* circumstances by adopting piecewise linearisation (see p 143). Piecewise linearisation may be used to model electricity transmission losses, which, based on a DC power flow model, are quadratic on duty (*ie*, exergy flow).

Plant behaviour may also be *arbitrarily dependent* on its immediately previous state — with this dynamic represented using (sets of) process state-transformation equations. These equations permit intertemporal effects to be included — for instance, thermal storage exergy loss over time.

Finally, the network capacity limits are taken to reside within individual plant — with the cut-in limits represented using (sets of) process threshold equations. The mathematical description of plant is covered in more detail on p 138.

Thermal sub-networks

Groups of plant exchanging heat are classified as thermal sub-networks — examples include district heating schemes and solar-assisted water heating. In this case, forced convection is treated as a heat rather than a mass flow problem, so long as the recirculation period is short compared with the duration of analysis. It is necessary to solve the intensive state of all thermal sub-networks prior to optimisation — which includes determining the temperature and phase ratio (*eg*, dryness fraction for steam) for the circulating transfer media at each port. This information is then used to create a single ′logical′ heat transfer interconnection for subsequent flow optimisation (logical in the computer science sense, see p 175).[22] To ensure tractability, certain restrictions are placed on plant interaction and behaviour, but these restrictions are typically acceptable in practice.

Intertemporal variability and dedicated storage

An exergy-services network is generally dependent on ambient conditions — which may, for instance, influence conversion plant performance, and/or renewable supply offers. Moreover, a network specification may change with time — for example, demand for service typically varies by time-of-day. Such variability gives rise to the idea of discretising one-year (say) into one-hour intervals (say) and sequentially optimising for each interval over the selected time-horizon.

This variability also presents the prospect of placing dedicated storage within the network. The simplest storage management strategy is to replenish using zero-selected cost offers whenever available, and discharge if this represents the lowest selected cost supply option — noting that the selected cost is that defined by the

[22] The *deeco* documentation uses the term ′net-enthalpy flow′ to describe this form of exergy flow.

optimisation goal. If future variability can be anticipated, at least to some degree of certainty, then a more sophisticated management regime involving withheld discharge and nonzero selected cost replenishment may be indicated.[23]

The integration model outlined thus far could be described as sequential time-local optimisation with waste recovery and non-anticipatory storage.

Decision support

The purpose of undertaking the type of network optimisation described here is to help support operational and design decisions. Design as an activity is taken to *include* exergy-services demand modification initiatives.

As noted previously, the network outcome sought is improved exergy-services sustainability. No single flow-dependent cost can fully proxy this goal, so the strategy best adopted is to evaluate several criteria in the hope that at least one investigated solution will score well across the board. If this does not eventuate, then the methods presented here will be of little use. The sustainability costs considered in this thesis include: depletable fuel use, net CO_2-e release, and network-local impact causatives such as sulphur and PM_{10} emissions.

The degree to which a network can be modified depends, at least, on the depth of intervention under consideration. Four time-frames are often used to indicate this degree of depth, although their temporal interpretation is becoming less relevant. The following treatment assumes that pro-actively managed storage is *not* present in the network.

The *operational* problem is to identify an optimal constellation of invoked plant, with regard to its efficiency, consequent impact, and ability to source — in terms of the chosen set of cost criteria — yet subject to the constraints of existing demand profile and fixed infrastructure.

The *short-run* problem is to seek an optimal constellation, as above — but in this case demand profiles may be modified using short lead-time mechanisms.[24] Demand shifting opportunities can be investigated by inserting factitious electricity storage immediately upstream of electricity demanding plant and analysing the resulting dynamics.

Several demand modification options present in the short-run, some of which require

[23] One form of intertemporal storage problem receiving attention in New Zealand is that of multi-dam hydro-reservoir optimisation. In this case, the main source of uncertainty is hydrological, and the principal goal is revenue maximisation rather than improved sustainability. In addition, the management regime will be tempered by caution if recent profitability has been poor. (Yang and Read 1999).

[24] Short and long-run are used in their economic sense, in that certain factors can be deemed fixed in the short-run, but varied in the long-run. For exergy networks, the infrastructure is taken to be static in the short-run.

real-time information about the condition of the network. First, changes in consumer tariffs influence short-run demand profiles, at least to some degree, although this point of intervention is upstream from point of exergy-service demand. Thus, time- and place-of-use differentials and peak load charging encourage demand shifting, and aggregate price rise tends to suppress aggregate demand. Market liberalisation (at least by stated intent) endeavours to provide monetary cost-reflective pricing signals, but for reasons of allocative efficiency rather than improved sustainability.[25] Second, non-time-of-use-sensitive customer-domain equipment can be placed under the authority of the grid operator, usually in exchange for lower tariffs — a scheme known as 'controllable load'. Finally, allocation under inescapable constraint is somewhat different in context and necessitates rationing, based on, for example, non-interruptibility premiums or some other predetermined degree-of-need priority. The optimisation method under discussion cannot directly support price-based demand suppression or *refusal to supply*, so such interventions, if required, will need to be included prior to formulating the model.

The *long-run* problem is to tackle additionally the remaining problem constraints — through infrastructure renewal and/or consumer preference modification. Infrastructure renewal includes plant retirement, upgrades, additions, and new interconnections, and also covers the customer-domain. Consumer preference modification techniques include public education and commercially-motivated advertising.[26] The infrastructure problem can be evaluated directly by including a number of new infrastructure options and re-running the optimisation procedure — this then naturally selects the plant which perform best in context and leaves idle those which do not.

The *very-long-run* problem captures technological advance, noting that nowadays technology cycles often undercut the economic life span of plant. Near-term developments can often be anticipated and projected forward, preferably with some margin for development risk. In particular, the performance/price trajectory for a number of emerging distributed technologies (*eg*, photovoltaics and fuel cells) is showing strong improvement.[27]

Figure 1.2 summarises the preceding discussion by depicting the network optimisation problem using three main components. Each component can be addressed to a differing degree, depending on the depth of intervention selected and other solution constraints. However, irrespective of the solution scope, the exergy-services sus-

[25] Setting aside the fact that achieving 'allocative efficiency' within network-mediated markets is highly problematic (Harvey and Hogan 2000).

[26] For example, domestic electricity demand in New Zealand dropped 14% during the 1992 hydro-crisis following public appeals to conserve. Few people reported a noticeable reduction in living standards and some social benefits may have accrued.

[27] The strength of positive feedback between uptake and development, particularly in regard to new renewables, is a major point of contention amongst econometric-based policy modellers. See, for example: Koomey *et al* (1998).

tainability problem requires that all three components be tackled in an integrated fashion.

The method given does not explicitly optimise the efficiency characteristics of individual plant, but rather the way in which these plant work together in context of location-specific demand *and* their connectivity *and* the potential contribution from lower impact sourcing. The method therefore depends on a sufficiently precise representation of the ′physical′ and ′informational′ structure of the network under consideration. This structure needs to be identified first hand and at sufficiently high resolution, and cannot be recreated satisfactorily using statistical decomposition techniques applied to low resolution data.

Furthermore, networks often exhibit counter-intuitive behaviour due to what can be described loosely as ′network-effects′ — these arise from the discrete (*ie*, network-based) and nonlinear (*ie*, capacity limited) nature of the underlying system. This means a small change in demand may result in the emergence of a radically different (and possibly unexpected) optimal routing pattern.

Monetary cost trade-off

Monetary cost considerations apply to real problems, so that the sustainability-based network integration techniques outlined here would normally be used in conjunction with some standard project appraisal technique, such as discounted cash flow analysis (DCF). In which case, the sustainability benefits of proposed changes could then be compared with the financial benefits or costs of these changes.

Figure 1.3 shows a number of design options evaluated for the city of Würzburg and plotted comparatively (as black triangles). A ′trade-off line′ has been added (in grey) — generally only those options which fall on the lower right corners need further consideration (the study which produced these conclusions is described more fully on p 160).

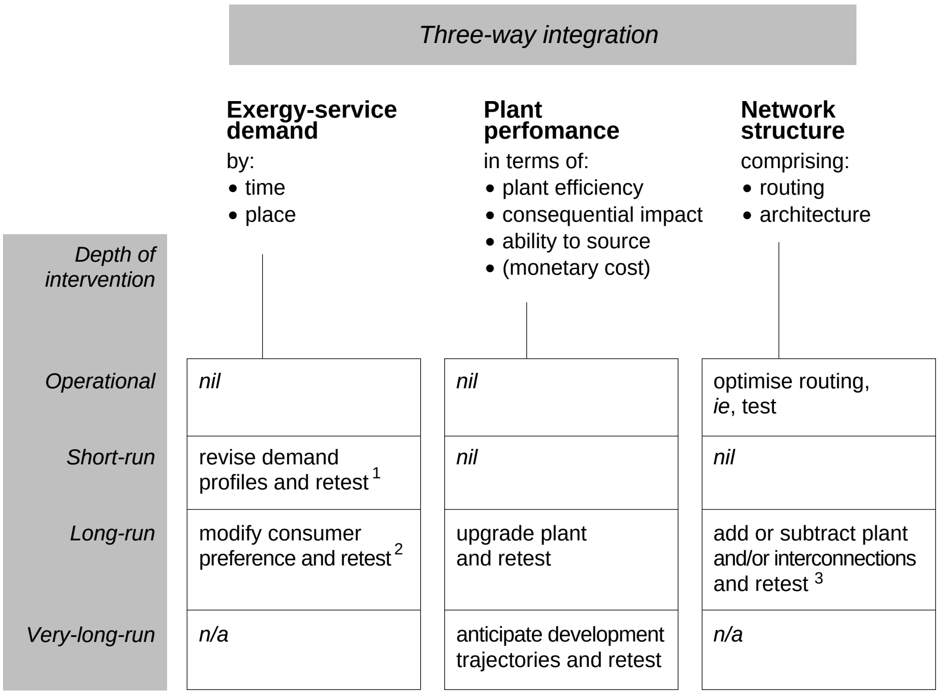
The question of whether a series of small design increments or a single larger intervention would be more beneficial can also be traversed using the model.

Sustainability outcomes

In terms of the overarching goal of this thesis, no built system is likely to be unambiguously sustainable. Nonetheless, every reduction in depletable fuel use rate, net CO_2-e release rate, or local environmental impact should improve prospects for the future.[29]

[28] Source: Bruckner, Groscurth, and Kümmel (1997, fig 2, p 1010), re-captioned.

[29] The nature and pace of any transition towards a more sustainable society ultimately rests with citizens, in accordance with the extent of their influence. Such progress will need to be ethically informed, as neoclassical economic self-interest alone cannot provide sufficient incentive (Faucheux,



Three-way integration

Depth of intervention	**Exergy-service demand** by: • time • place	**Plant perfomance** in terms of: • plant efficiency • consequential impact • ability to source • (monetary cost)	**Network structure** comprising: • routing • architecture
Operational	*nil*	*nil*	optimise routing, *ie*, test
Short-run	revise demand profiles and retest [1]	*nil*	*nil*
Long-run	modify consumer preference and retest [2]	upgrade plant and retest	add or subtract plant and/or interconnections and retest [3]
Very-long-run	*n/a*	anticipate development trajectories and retest	*n/a*

Notes: 1. Includes collective-interest control, tariff revision, and explicit rationing.
2. Includes public education and commercial advertising.
3. Includes waste recovery opportunities.

Figure 1.2 *The three components of the exergy-services sustainability optimisation problem. Each aspect can be addressed using the strategies indicated, depending on the depth of intervention selected. Proposed changes need to be numerically modelled to establish their merit or otherwise, hence the use of the word 'test'. This diagram should be read in conjunction with fig 1.1. The diagram could also be extended to include investigation into storage management policy improvements.*

The evolution of a particular ESSS tends to be influenced by past monetary and non-monetary investments, by current and anticipated costs and revenues, and by the existing and projected regulatory and institutional context.

The types of networks under consideration can be either individually managed or market-based and multi-participant. In the latter case, the design of pricing incentives and/or institutional arrangements which enable or promote optimum changes in participant behaviour and network re-investment is extremely difficult. And although this issue is vitally important, lack of space precludes it receiving more than passing attention in this thesis.

Decision-maker jurisdiction and network boundaries

The boundaries of a network under examination are generally set to coincide with the jurisdiction of the decision-maker requesting the modelling. Therefore, the optimisation model under review could usefully be applied to the following problems:

- **remote-area schemes** — remote collectives may find grid-independence financially attractive,[30]
- **industrial sites** — an operator may wish to demonstrate baseline (*ie*, post-1990 CO_2-e business-as-usual) reductions in order to claim emissions credits (assuming a certain form of tradable permit is introduced) and/or manage their ongoing greenhouse gas emissions,
- **municipalities** — a city administration may wish to investigate infrastructure planning options and/or user incentives,
- **national policy** — central government may wish to evaluate energy-services supply system sustainability policy options using quantitative modelling.

So whilst the overarching system remains the ESSS, the division of the ESSS into autonomous networks as described could be interpreted as ′distributed planning′.

Patchworked networks and import/export

It would be rare for the various applications listed above to be fully spatially isolated — that is without any form of external supply and/or demand connection. Indeed, most would be gatewayed to abutting exergy networks, if only on a batch rather than a continuous supply basis. The term ′patchwork′ is therefore used to indicate that networks may be either *explicitly* interconnected through gateways which allow for the direct exchange of exergy, or *implicitly* interconnected by supplying, or being

O'Connor, and van der Staaten 1998, Howarth and Norgaard 1995).

[30] In New Zealand, the legal obligation for electricity grid operators to connect expires in April 2013 under s 62(6) of the Electricity Act 1992. Lines companies typically cost rural lines in the order of $ 1000/km *pa* to maintain and operate. As a consequence, many rural communities are now anticipating the need to establish their own electricity grid-independent ESSS schemes.

in a position to supply, exergy-services to the same point of service-demand. The sum of patchworked networks at any given time makes up a single ESSS.

Furthermore, the principal network need not be physically contiguous, but nonetheless remains implicitly joined if future interconnection possibilities — either at the point of demand or back upstream — exist.

The model under discussion supports the import and export of exergy with neighbouring (*ie*, gatewated) networks, given that restrictions similar to those applied to plant within the principal network are met — namely, that the characteristics and costs of the imported or exported exergy can be identified in advance and that these characteristics and costs are independent of the chosen flow. This equates to the assumption that the operating regime prevailing within the principal network does not affect the cost structures in the wider system.

Exported exergy reduces network-wide costs by an amount equal to that which would arise from an equivalent import. In the absence of a more sophisticated storage and export management policy, exergy is only exported if surplus, after storage replenishment, and available at zero selected cost.

To recap, the unit of optimisation is the network whose boundary is determined by the jurisdiction of the decision-maker in question.

Network redesign

The Würzburg study found that, given a fixed set of time-series demand profiles, some clusters of technologies will reinforce, and others, which may have earlier seemed promising, will counteract. Therefore, intuition provides a poor guide for decisions concerning network plant utilisation (*ie*, routing) and re-investment (*ie*, architecture) — indeed, optimisation modelling is generally taken to be a better strategy for guiding network operation and design initiatives.

Notwithstanding the above observation, intuition is required to identify potential opportunities for design improvement. Setting aside the issue of monetary cost-effectiveness for the moment, several strategies present themselves in terms of this endeavour:

- **exergy quality mismatch** can provide useful leads, particularly when working back upstream from a point of exergy-services demand,[31]
- **plant efficiency improvement** *per se* at any point in the network is likely to reflect positively on the system-wide sustainability objectives (assuming that most sustainability costs are flow-related rather than embodied),
- **renewable sourcing** will often, but not always, displace depletable sourcing,

[31] Exergy quality is discussed on p 65.

- **demand modification strategies** and/or **storage opportunities** may become apparent after inspecting plant utilisation statistics.

Figure 1.4 summarises these ideas in a somewhat simplistic form.

The application program deeco

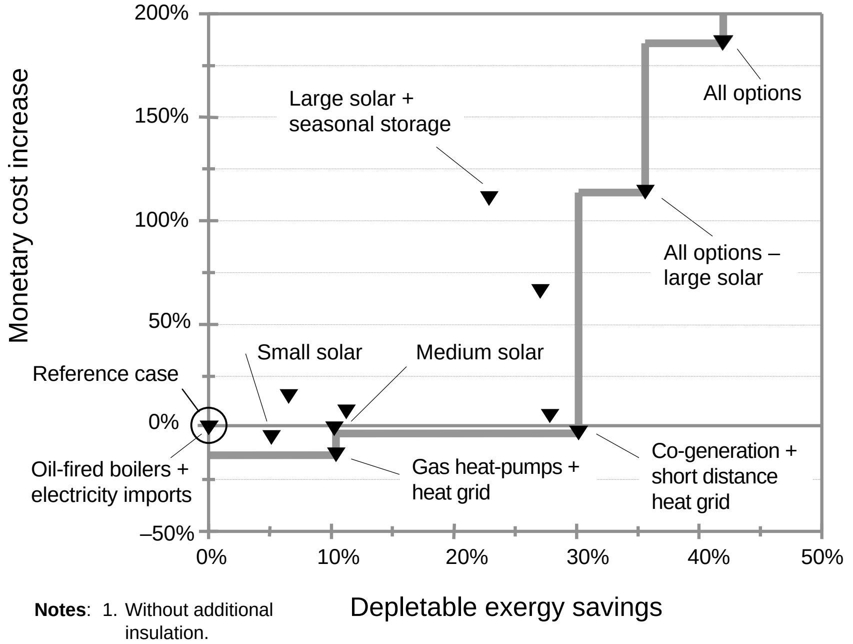
The application program *deeco* provides a numerical modelling environment for undertaking ESSS network integration of the type under consideration. *deeco* contains built-in support for a number of common exergy processes and several sustainability-based flow-dependent costs. The program was coded by Dr. Thomas Bruckner, as part of his PhD research, in collaboration with Professor Reiner Kümmel and Dr. Helmuth Groscurth. The first release was completed in late-1995.

For a given scenario, *deeco* reads in the network specification including time-series demand profiles and ambient conditions plus flow-dependent and flow-independent cost information, and writes out a series of least-cost routing solutions and selected aggregate statistics. If the built-in process modules are not suitable, new process modules can be written and added to the program. Data interchange is via suitably structured ASCII text files, which can either be prepared and judged manually, or generated and interpreted with the help of a custom database application. *deeco* has primarily been structured to deal with municipal problems, and so may require adaptation for other classes of problem.

By default, *deeco* steps through one year in 1 hour intervals to give 8760 network optimisation calculations. This combination of resolution and time-horizon is generally sufficient to capture both daily and seasonal variations. Explicit optimisation across time-intervals is not performed, but surplus renewable exergy capture and waste recovery can be carried over to the next interval whenever spare storage capacity exists.

deeco is, in essence, a data abstraction that approximates and overlays a real physical system. Plant behaviour is represented by mathematical statements. Data indicating the external conditions, the exergy flow intensive states, the sought demand, and the consequent costs is transferred between interconnected processes and these processes and their surroundings, as the program proceeds. During the time-local optimisation step, the flow of demand data is anti-parallel to the flow of ′exergy′.

The task of transferring *deeco* from an ageing Hewlett-Packard workstation to a contemporary Intel-compatible computer made up a significant component of this thesis. The work undertaken to achieve this ′port′ is described in chapter 2.



Notes: 1. Without additional insulation.

Figure 1.3 *Results from the Würzburg study with a trade-off line added in grey. Generally, only those options which fall on the lower right corners need further consideration.*[28]

'Sustainable energy'

=
Plant efficiency
+
Renewable sourcing
+
Demand management
+
Storage management
+
Waste recovery
+
Optimal routing

Figure 1.4 *The central theme of this thesis restated in simple terms. 'Sustainable energy' is taken to be shorthand for improved exergy-supply system sustainability.*

Model structuring

Problems of the general type tackled by *deeco* are often framed using matrix algebra and therefore optimisation is described in terms of matrix manipulation. Alternatively such problems can be framed using graph structures (see p 86) with optimisation described in terms of augmenting path searching. The graph-theoretic description is adopted in this thesis because it more readily captures the underlying reality, because it is used to solve thermal sub-networks and undertake storage replenishment and export, and because it facilitates abstraction in the program domain through use of object-oriented techniques. *deeco* itself, however, uses matrix techniques rather than graph algorithms for the actual optimisation procedure.

Research fields

This work draws upon the following topics and disciplines:

- open-systems thermodynamics (physics)
- ′energy′ conversion and exergy analysis (engineering)
- dynamical systems theory and process control theory (mathematics)
- graph theory (mathematics and computer science)
- mathematical programming and optimisation (operational research)
- software design and topological programming methods (computer science)
- impact assessment (environmental science)
- production theory and related topics (microeconomics)
- sustainability theory (public policy)

As can be seen, the list is broad, but probably not fully complete. This sweep of disciplines both facilitates insight and raises difficulties — and gives ′energy′ modelling its conceptual appeal.

1.2 General information

Engineering sign convention

An engineering sign convention is used with regard to thermodynamic systems: heat *in* and work *out* are taken as positive. Similarly, exergy *out* is taken as positive.

Nuclear power

This thesis expressly excludes nuclear fission technologies. The reasons are threefold. First, generalised statements about exergy systems become more cumbersome if mass-consuming fission processes are included.[32] Second, nuclear power stations are not financially viable under current cost and risk structures (*The Economist* 1995). Third, nuclear power production is prohibited under planning law in much of New Zealand. Therefore, all comments regarding exergy processes and asserting generality exclude nuclear fission.[33]

1.3 Key literature

Two documents are so central to this thesis that they warrant specific mention in the introduction. One is Dr. Bruckner's PhD thesis, from which key sections have been translated into English. The other is a paper arising from work supporting Dr Bruckner's thesis and published in the journal *Energy*.

> Bruckner, Thomas. 1997. *Dynamic energy and emissions optimization for regional energy systems (part translation).* PhD thesis. University of Würzburg, Germany.[34]
>
> Groscurth, Helmuth-M., Thomas Bruckner, and Reiner Kümmel. 1995. Modeling of energy-services supply systems. *Energy.* **20**(9):941–958. ISSN 0360 5442.[35]

Both documents complement the material in this thesis, and both are recommended to interested readers. Details of other *deeco*-related material are given in chapter 8.

1.4 External relations

This thesis relied on assistance and interest from a number of individuals from outside the University of Otago. In particular, the following people kindly provided valuable information and feedback:

- Dr. Thomas Bruckner, Institute for Energy Engineering, Technical University of Berlin, Berlin, Germany.

[32] In this case, mass-energy balances are required.

[33] If required, nuclear power can be included in *deeco* using the heat equivalent method, as given in definition 5 in Bruckner (1997, translation).

[34] The translation is being proofed at the time of writing — with the version number currently at 0.3. Copies may be requested from the writer, but their release would need approval from Dr. Bruckner. The original document is cited as Bruckner (1997) and the translation is described as Bruckner (1997, translation) in the body of this thesis.

[35] The abstract reads: "A general model framework is developed which describes regional and municipal energy systems in terms of data-flow networks. It provides a highly flexible tool for dynamic and stochastic minimization of primary energy demand, emission of pollutants, and monetary cost. Included are conventional energy-supply techniques, rational use of energy via heat-exchanger networks, heat pumps, and co-generation, demand-side measures such as insulation of dwellings, and utilization of renewable energy sources."

- Scott Dunavan, Topoclimate South Trust, Mataura, New Zealand.
- Matthew Everett, Energy Efficiency and Conservation Authority, Wellington, New Zealand.
- Karen Garner, Applied Mathematics, Industrial Research Ltd, Wellington, New Zealand.
- Dr. Helmuth Groscurth, Hamburg Electricity Works, Hamburg, Germany.
- Dr. Jonathan Lermit, Transpower NZ, Wellington, New Zealand.
- Dr. Dietmar Lindenberger, Institute for Theoretical Physics, University of Würzburg, Germany.
- Molly Melhuish, Independent Analyst, Wellington, New Zealand.
- Professor Richard Nowakowski, Mathematics, Dalhousie University, Canada.
- Associate Professor Hugh Outhred, School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia.
- Dr. John Peet, Department of Chemical and Process Engineering, University of Canterbury, Christchurch, New Zealand.
- Dr. Grant Read, Energy Research Modelling Group, Department of Management, University of Canterbury, Christchurch, New Zealand.
- Conal Tuohy, ReddFish intergalactic,[36] Wellington, New Zealand.
- Philip Walton, Operations Research Analyst, USA.

During the course of the project, Dr. Groscurth, who was instrumental in the development of *deeco*, visited the Physics Department for 4 weeks, from late-October to late-November 1998.

Some of the ideas contained in this thesis have been presented at the following forums:

- Keith Dawber Memorial Seminar, New Zealand Wind Energy Association, Dunedin, 27 July 1998 (Morrison 1998).
- Sustainable Energy Forum, Auckland, 24–25 June 1999 (Morrison 1999).[37]
- Ministerial Inquiry into the Electricity Industry, written submission dated 13 March 2000, and OHPs dated 31 March 2000.[38]
- Electricity Engineers' Association of New Zealand conference, Auckland, 16–17 June 2000 (Morrison 2000).

[36] ReddFish intergalactic is an MS Windows platform programming consultancy.

[37] A corrected version is available from: `http://www.physics.otago.ac.nz/robbie/sef99.html`.

[38] This material is available from: `http://www.physics.otago.ac.nz/robbie/elecInq.html`.

1.5 Closure

This chapter presents two concepts that are rare in the exergy-services sustainability debate. The first concerns the systematic integration of exergy-services supply networks — a lack of interest (setting aside specialist work concerning the technical integration of generation technologies into electricity grids) which is remiss given that distributed technologies, including new renewables, are particularly sensitive to network placement and demand matching. The second is the notion that exergy-services supply systems should be explicitly optimised for sustainability rather than first law efficiency. These two concepts, when combined, form the basis of the network integration application program *deeco*.

Network redesign includes exergy-services demand, and hence overt exergy-services demand modification is taken to be a legitimate response to the exergy-services sustainability problem. The effect of demand modification in terms of time, place, and magnitude can be readily evaluated using the model presented. □

Chapter 2

Programme of work

This chapter outlines the programme of work undertaken as part of this thesis. In particular, it describes the work required to port deeco *from a circa 1991 Hewlett-Packard workstation to a contemporary Intel-compatible desktop computer.*

2.1 Programme of work

Research proposal and early investigations

The first research proposal was submitted to the Physics Department in December 1996, and carried the title: *End-use energy decisions : seeking more environmentally sustainable outcomes within deregulated market environments.* After revision, the proposal was accepted in February 1997 (issue C). Work began in earnest in July 1997 under the supervision of Professor Gerry Carrington.

The principal research goal was to identify a set of universal guidelines which would assist motivated energy users to make plant and demand decisions favouring sustainability. This line of inquiry was not pursued, instead being replaced by the view that situation-specific modelling provides a generally better strategy — particularly where the technologies and demand patterns involved could exhibit significant synergy or counteraction.

Apart from this change of tack, most of the analytical framework suggested for viewing energy systems and consequent impacts withstood scrutiny, and was adopted and developed. But whilst the concept of exergy-services supply networks played a central role in the proposal, the writer was not aware then of the existence of numerical techniques which could be used to cost-optimise flow networks — these numerical techniques now constitute a central component of the project.

Early activity concentrated on defining the concept of environmental impact, seeking proxies for environmental impact relevant to exergy systems, and establishing methods for identifying and analysing marginal exergy-services supply chains. A marginal supply chain is that which is invoked to supply the last increment of a particular demand.

With regard to marginal supply chain analysis, two useful leads came to light. The first was a paper by Groscurth *et al* (1995) describing the model framework NEMESS upon which *deeco* is based. The second was a paper by Ossebaard, van Wijk, and van Wees (1997) making reference to the ENERPAC suite of thermodynamic analysis routines.

During October 1997, the writer contacted Dr. Groscurth who readily arranged for Dr. Bruckner to forward the C++ source code for *deeco*. In November 1997, the writer contacted Dr. Nieuwlaar, Utrecht University, and obtained the well-documented ENERPAC 5 routines (see p 176) as ANSI C source code at nominal cost (Nieuwlaar 1996).

deeco had the clear advantage over marginal supply chain analysis in that it explicitly factors in the network structure and undertakes whole system optimisation.

Porting deeco

For those unfamiliar with programming, source code is written in a high-level programming language (in this case, C++) and submitted to the system compiler to produce an executable program for that particular platform (*ie*, combination of operating system and hardware). If a different platform is to be used, then the source code will need to be recompiled for the new platform — an exercise known as ′porting′. A number of factors determine whether the porting process will be trivial or otherwise. For programs with a command-line interface (CLI) (*ie*, *deeco*) the issues include: differences in language implementation between the two respective compilers, differences in operating system calls, the presence of specialist libraries, and whether platform-specific features (*eg*, hardware-level exception handling) prove problematic. For the most part, standardisation efforts to make CLI C++ application programming source compatible across common platforms have not been successful (on the other hand ANSI C / POSIX–1 [1] compliant C source code is regarded as reasonably portable.)

Dr. Thomas Bruckner wrote *deeco* as part of his PhD research while at the Institute for Theoretical Physics, University of Würtzburg, Germany. *deeco* was developed on a *circa* 1991 Hewlett-Packard (HP) RISC 700 series workstation.[2]

deeco comprises 18 000 lines of code in 21 modules (*ie*, `*.h` files) and uses some

[1] POSIX–1 is IEEE Portable Operating System Interface standard 1003.1 covering systems calls.

[2] The HP 700 processor was many times faster than the Intel 486 chipset current at that time.

66 user-defined classes with 3-deep inheritance.

Upon receipt, the *deeco* source was examined, and a list made of system and library calls and other environmental requirements. Two experienced C++ programmers looked over the code, but neither was willing to pass judgement as to its fitness.

Nonetheless, during late-1997, a decision was taken to use *deeco* if at all possible. This decision was not without risk as a suitable platform had not yet been secured. Attention then turned toward identifying options for porting *deeco* to a contemporary Intel-compatible computer. *deeco* was coded in *circa* 1990 C++, called obsolete compile-time libraries known collectively as USL Standard Components 3.0, and handled arithmetic errors (*eg*, div-by-zero) at the program rather than hardware level.

In mid-February 1998, Dr. Chris Handley, Computer Science Department, agreed to act as a co-supervisor. Dr. Handley's involvement provided additional depth to a project becoming increasingly involved with computer science issues.

Source code for USL Standard Components was located at a price of USD$ 1500, but the vendor warned that transferring this code to a modern compiler would be arduous. Further investigation revealed that SCO UnixWare 2.1.2 SDK / x86 operating system (abbreviated as UW2.1) and software development environment (SDK) supported the problematic libraries, and should meet all other specifications.[3] UW2.1 is based on the System V UNIX lineage. See p 154 for a review of the two environments.

Consequently, a 200MHz Pentium Pro compatible computer was purchased and loaded with both Windows NT 4.0 and a non-commercial copy of UW2.1 on 19 February 1998. An updated screen device-driver was required and was located, downloaded and installed. The monitor failed and was repaired. `NEdit`, a well-featured open-source text editor, was downloaded, compiled, and tested. Other tools, including a PERL interpreter were added from the SCO shareware CD. By the end of April, the system was considered usable for development. Sometime later, the FAT16 partition was mounted in order to transfer large files to the Windows system. Malcolm Fraser, Physics Department, added the dual-boot utility `OSBootSelect` to facilitate operating systems changeovers.

The *deeco* source code had earlier been copied across during March 1998. Details of the debugging do not warrant inclusion at this point, except to note that 11 distinct issues were encountered. During this process the system graphical debugger `gdb` was configured to suit requirements and a number of KornShell scripts were written to automate repetitive tasks. Dr. Richard O'Keefe, Computer Science Department,

[3] SCO is a major software vendor. The term SCO is a contraction of Santa Cruz Operation.

made an invaluable contribution by assisting with porting modifications.

A small tutorial program using the USL Standard Components graph libraries — based on material from the SC manual (Weitzen 1992) — was coded and tested. Building, debugging, and single-stepping this program, named `widget`, provided vital information about the compiler switches required for *deeco* and also gave insight into the construction of the graph libraries. A listing for `widget3` is given on p 200. Several other small C++ programs were written to test floating point exception handling options and other porting-related issues.

A clean compile was obtained for *deeco* on 31 August 1998, but the resulting executable duly exited after failing to locate data files. Dr. Bruckner had earlier forwarded a trivial data-set for testing purposes, named `testdata`, together with output obtained from his HP platform. The trivial data set was successfully trialled on 02 September and the output was numerically identical. Dr. Bruckner then forwarded a large data-set, named `testwue`, and this was run on 05 November using a non-optimised executable.[4] This scenario took 0.7 hours — as opposed to 1.2 hours on the original HP workstation. Because the two platforms differed in processor architecture and the two executables differed in their floating point exception handling, output was not identical. A PERL script was written to compare sets of results and highlight discrepancies for further investigation. Filtered output from the large data-set was carefully reviewed by Dr. Bruckner on 06 November and all the variations were ascribed explanations. Dr. Handley also regarded the discrepancies to be acceptably close under the circumstances.

The elapsed time from source code download to program verification was 11 months.

SCO ceased to support UW2.1 in mid-1998. Concern over the use of a stranded development environment prompted a decision to trial SCO UnixWare 7.0 UDK / x86 (abbreviated as UW7.0). This software was purchased on an academic licence and Malcolm Fraser, Physics Department, loaded it onto a 300MHz Pentium Pro-compatible machine on 05 November 1998. The *deeco* UW2.1 source code unexpectedly failed to compile, due to problems with library header files. To date, no attempt has been made to determine the cause or seriousness, or install the older SDK compiler as a work-around. Nonetheless, the UW2.1 executable runs directly on UW7.0 without problems.

Development work on *deeco* was halted at this point, as the Intel port had been successful. Furthermore, several important conclusions had been established:

- *deeco* can be developed and run on an Intel-compatible PC using UW2.1 / SDK,
- *deeco* compiled under UW2.1 / SDK will run directly on UW7.0 — an operating system which promises reasonable longevity and bundles web support,

[4] Optimised executables take longer to compile and are generally avoided during development.

- *deeco* may well compile directly on UW7.0 / UDK or, alternatively the older SDK compiler can be custom loaded onto UW7.0.

The modified source code was returned to Dr. Bruckner on 30 November 1999, so that his colleague Dr. Lindenberger could establish *deeco* on an Intel-compatible PC. Dr. Lindenberger successfully recompiled this code on a 450MHz Pentium III machine running UW2.1 during January 2000.

Translation

In parallel with the *deeco* port, salient parts of Dr. Bruckner's PhD thesis were translated from German. Jan Diettrich, Physics Department, dictated onto tape, and the writer transcribed this material and reformatted it using Dr. Bruckner's original LaTeX files.[5] The translation comprises 50 pages and provides invaluable information about the details of *deeco*. The translation is cited in full on p 20.

Network algorithms

Attention then turned toward achieving a working knowledge of the numerical theory underpinning *deeco*. Dr. Robert Aldred, Mathematics and Statistics Department, allowed the writer to attend graph theory lectures presented under MATH 340.[6] The shortest path and out-of-kilter graph algorithms, given by Smith (1982) as Pascal listings, were translated into MATLAB 5.3 and trialed. Graph definition and traversal algorithms from Gibbons (1985) were similarly coded and trialed. Sample simplex linear programming (LP) routines were downloaded from the MATLAB website and tested, and MATLAB's `linprog` LP routine, contained in the Optimisation Toolbox, was investigated. A representative program listing is given in appendix D.

Graphical user interface (GUI) network definition and display methods were also explored using the 2-D graphics capabilities of MATLAB. This work, although unfinished, looks promising and MATLAB is certainly a reasonable environment for prototyping GUI concepts of this nature.

Deployment options

In addition to the programming efforts outlined, development and deployment options for *deeco* were investigated. Much of this work was undertaken in collaboration with Dr. Groscurth while he was in New Zealand. The writer and Dr. Groscurth interviewed several senior programmers, including Conal Tuohy and Scott Dunavan, who provided valuable suggestions and opinions — particularly in relation to making *deeco* available to potential users via a web interface (see p 156).

[5] LaTeX is a document preparation system.

[6] The graph theory relevant to this thesis is covered in section 6.4 (see p 85).

Scenario development

Whilst no actual modelling using *deeco* was attempted, several possible New Zealand case studies were reviewed and a national policy scenario selected for future development — this scenario is outlined in chapter 9. *deeco* was originally developed to model municipal systems and will undoubtedly require modification for use in a national model. Potential solutions and work-arounds have been discussed with Dr. Bruckner.

2.2 Closure

The original research proposal allowed for numerical modelling, namely: "the option also exists to develop rudimentary computer simulations." The writer was extremely fortunate to locate *deeco* and obtain the source code, and, furthermore, to gain the interest and support of Dr. Bruckner and Dr. Groscurth. *deeco* added immeasurably to the depth of the thesis.

And although the writer had taken undergraduate courses in numerical methods, C++ programming, and abstract data structures, the overhead required to understand UNIX system administration, the UNIX application programming environment, algorithmic graph theory, and LP optimisation was still considerable. □

Chapter 3

The challenge of sustainability

This chapter discusses sustainability — both generally and for exergy-services supply systems — and the selection of suitable cost functions for use in ESSS optimisation.

3.1 Introduction

The sustainability crisis is a recent phenomenon, having only attracted significant attention in the last 30 years. Exergy-services supply systems in their current form are a major contributor to the crisis. But fortunately alternatives based on renewable sourcing, systematic integration, and a more equitable distribution of access to exergy-services offer the prospect of more sustainable exergy-service provisions, even in the face of significant population growth.

The sustainability crisis can be framed as a generalised depletion issue. On the one hand, material resources are being used at a faster rate than replenishment. And on the other, waste is being discharged more quickly than relevant local and non-local biophysical systems can cope.

ESSSs should be optimised for improved sustainability, and, ideally, the cost functions used to proxy for sustainability should be identified in some systematic way. However, no such generalised selection methodology exists, so the rationale by which ESSS optimisation cost functions should be selected provides the theme of this chapter. The cost functions explicitly supported in *deeco* are listed elsewhere (see p 134).

3.2 Sustainability in general

Assimilative and evolutionary capacity

Important biophysical processes and, more latterly through biodiversity loss, evolutionary processes are being compromised as a result of cumulative human activity.

This activity is occurring with little apparent regard for the likely long-term and long-range adverse consequences — consequences which are, arguably, beginning to present as serious and intractable systemic problems. (Brown *et al* 1999, Ehrlich and Holdren 1988, United Nations Environment Program 1999).

The ability of receiving environments to cope with waste streams is known as ′assimilative capacity′. Receiving environments are difficult to model — they typically exhibit nonlinear and time-lagged dose/response characteristics for a single waste stream, and are often subject to multiple waste streams. Assessing the cumulative impact of multiple waste streams and then setting suitable discharge entitlements is highly problematic — consequently some authors recommend a substantial margin be included to allow for ′irreducible uncertainty′ (May 1986, Parliamentary Commissioner for the Environment 1998). And although it is not possible to fully predict the dose/response behaviour of receiving environments, assimilative capacity is increasingly being viewed as a resource which should be priced in some form.

The non-local issue attracting most concern at this point is the climate crisis. As anthropogenic greenhouse gas release rates rise, so too do their atmospheric concentrations — this in turn intensifies the radiative forcing of the climate system and leads to the modification of weather and climate patterns. Extreme event frequency, severity, and predictability are expected to deteriorate, and the sea-level is expected to rise. (Intergovernmental Panel on Climate Change 1994, IPCC website[1]).

A number of industrialised countries are investigating institutional measures in order that anthropogenic CO_2-e release presents as a direct or indirect cost to production and consumption. Such measures include dedicated carbon taxes, tradable emissions permit regimes, and/or restricted discharge entitlements.

Resource depletion

Depleting production inputs also poses a major problem. Some inputs are simply not substitutable, *eg*, phosphorus for fertiliser. Others, like fossil fuels, are, in theory, fully substitutable, but the switch to renewable sourcing may well be accompanied by high transitional costs and the stranding of some conversion and end-use plant.

In particular, crude oil extraction rates are projected to climax and decline within 10 years — a dynamic that cannot be overturned by pricing mechanisms and which may prove highly problematic for some sectors (Campbell and Laherrere 1998). At the time of writing, the price of crude oil had started to rise, but it is too early to determine whether this signal is meaningful or not.

[1] The web URL is: `http://www.ipcc.ch`.

The 'limits-to-growth' debate

Meadows *et al* (1974) published a controversial system dynamics-based investigation[2] into the consequences of current production, consumption, and population trajectories. Calibrating dynamic models of this type is always problematic, but even so the overarching conclusion appeared robust — material throughput (including fossil fuels), waste toxicity, and population levels should be urgently stabilised.[3] Supporters continue to argue that every opportunity should be taken to reduce the toxicity and magnitude of current production waste and consumer disposal streams (Jackson 1996).

Other analysts believe the sustainability crisis is largely illusory — that the planet is a place whereby human technical and economic ingenuity will ensure that food production, resource extraction, and pollution control can and will keep pace indefinitely with human wants (adapted from Peet 1992, p 157). Some localised interpretations of historical events lend qualified support for this so-called exemptionalist view, whilst others do not. Further, the degree to which past events can inform the current predicament is also under dispute.

Nonetheless, one view gaining credence is that a stepwise increase in the productivity with which society uses both material resources and exergy is required — one that would rival the increases in labour productivity that accompanied the industrial revolution.[4] (Hawken, Lovins, and Lovins 1999).

Resolution of the 'limits-to-growth' controversy is clearly beyond the scope of this thesis, but, interestingly, the *deeco* model will accommodate either perspective. Rather, it is the choice of problem and goal, the interpretation of numerical results, and the nature of the response envelope deemed legitimate that provides the contention.

Sustainability as an operational concept

Sustainability is defined here as an operational concept which requires choosing those behaviours, activities, and technologies which are less, or less likely to be, non-sustainable (adapted from Peet 1992, p 210–211). This definition shares conceptual similarities with the Pareto optimality test for allocative efficiency, a central tenet

[2] Forrester (1965) is taken as the starting point for this field of modelling.

[3] This study was later updated and released as Meadows, Meadows, and Randers (1992).

[4] 'Productivity' is the marginal physical product (MPP) of a particular factor input I_i as given by (Nicholson 1995):

$$P_i = \frac{\partial q}{\partial I_i} \qquad \text{with} \quad q = q(I_1, I_2 \ldots I_n)$$

where: q is the single product production function for the firm in question
I_i is the factor input [physical unit]
P_i is the marginal physical product [items/physical unit]

in neoclassical economics. Sustainability is, therefore, not an absolute concept, in the sense that it is unlikely to be fully achievable for contemporary societies.

A number of authors have attempted to systematically rank existing and proposed activities in terms of their sustainability. Contributions include: Ayres (1996), Clayton and Radcliffe (1996), Cocklin (1989), Dovers and Handmer (1993), Hannon, Ruth, and Delucia (1993), Robert, Holmberg, and Eriksson (1995).

Local impacts

Local impacts are more difficult than non-local impacts to factor into an optimisation model. Local impacts, should, in principle, extend uniformly over the entire geographic area of the network under consideration before they are included — in other words, waste stream *intensity* issues are not supported by the optimisation model under consideration. Hence, plant with highly localised adverse effects should be evaluated prior to inclusion — noting that this type of assessment is often required under planning law in any case. The collateral damage which arises from the construction and decommissioning of plant, rather than directly from its operation, is often highly local.

Exergy and ϵnergy as general proxies for biophysical impact

Researchers have proposed, variously, that flow exergy and flow ϵnergy can be used to proxy for environmental impact, at least as a first approximation. Contributions include: Kümmel (1989), Kümmel and Schüßler (1991), Rosen and Dincer (1997). Such methods have not been included as part of this thesis.

3.3 Exergy-services supply system sustainability

The network optimisation method presented in this thesis facilitates the search for an ESSS — either in terms of operation or redesign — which is *less non-sustainable*. Notwithstanding, the selection of a cost function or suite of cost functions to proxy for ESSS sustainability remains less than systematic. This section reviews the question of ESSS sustainability and issues surrounding the selection of suitable cost functions. These matters, as they apply to New Zealand, are revisited later in this chapter.

Depletable versus renewable sourcing

Exergy resources extracted from biophysical systems can be classified as either depletable or renewable. Historically, this division compared the extraction rate of the fuel stream with the biophysical replenishment rate of the source. More recently the classification has been extended to compare the rejection rate of any associated waste streams with the assimilative capacity of the receiving environment, at both local and non-local spatial scales.

The 1997 Kyoto Protocol set limits for the rejection rates of CO_2-e emissions for industrial countries (the Annex I signatories) from their agricultural and industrial systems. But in doing so, the Protocol legitimates their 1990 output with some second-order modification (the weighted-average reduction is −5.4%) whilst side-stepping the issue of Third World emissions track aspirations.

ESSS-related environmental impact

The UN Environment Program GEO 2000 report lists the following ′energy′-related environmental problem indicators applicable to industrialised countries:

Problem	**Indicator**	**Unit**
Climate change	Greenhouse gas emission rates	Mt CO_2-e / year
Acidification	Percentage of ecosystems in which acidity thresholds have been exceeded	%
Summer smog	Eight-hour average time-weighted 60 ppb ground-level ozone exceedence	ppm-hours
Urban air pollution	Percentage of urban population exposed to NO_2, PM_{10}, benzene, SO_2, and benzo(a)pyrene in excess of EU Air Quality Guidelines	%
Nuclear risk	Probable excess cancer fatalities due to nuclear power plant accidents in deaths per 10^8 population	number / year

Table 3.1 *′Energy′-related environmental indicators listed in United Nations Environment Program GEO 2000.*[5]

The problem classification shown in table 3.1 broadly aligns with that advocated in this thesis and specifically supported by *deeco*, except that problems associated with nuclear power are excluded. The issues of air quality may become more important, particularly if the uptake of certain forms of distributed generation continues.

But only one of the entries in the table — the CO_2-e emission rate — is a ′pressure′ indicator, with the remainder being ′response′ indicators.[6] The network optimisation model under discussion can only support pressure indicators, so that any issues represented by response indicators will need to be recast as flow-dependent costs prior to inclusion.

[5] Source: United Nations Environment Program (1999, p 352).

[6] The ′pressure/response′ terminology derives from the current New Zealand environmental policy analysis vocabulary.

Embodied costs and life-cycle analysis (LCA)

An ESSS may adopt renewable sourcing but the conversion plant (*eg*, wind turbine generators) still require and degrade material resources and will still embody environmental impact. As a consequence, life-cycle analysis (LCA) methodologies may need to be applied to ESSS assessment (Ayres, Ayres, and Martinas 1998).

deeco, in effect, undertakes a limited form of life-cycle analysis when it reports the network-wide amortised embodied costs. However, a wider LCA is required to establish the base information used by *deeco*. In addition, *deeco* can only directly optimise for the consequential (*ie*, flow-dependent) costs, although embodied (*ie*, flow-independent) costs can be addressed heuristically by adding and removing plant and evaluating the results (as is done in the case of monetary cost trade-off).

Cost selection methodologies

Multiple-criteria cost selection methods applicable to ESSS projects are reported in the literature, refer to: Bose and Anandalingam (1996), van Pelt, Kuyvenhoven, and Nijkamp (1990).

Cost-benefit analysis (CBA)

Cost-benefit analysis (CBA) techniques, which require all reasonably foreseeable cost and benefit streams to be monetarised and discounted prior to inclusion are, arguably, of little use in the ESSS sustainability problem. First, the problem of translating environmental and social costs into dollars using contingent (*ie*, non-market) valuation methodologies is highly problematic, particularly when those costs arise from something as uncertain as global change. Second, the selection of risk-adjusted discount rates is speculative at best. And third, the method is almost never linked to remediation, so that any impacted biophysical systems remain compromised.

3.4 Elected cost functions

The cost functions which appear generally to best represent the EEES sustainability problem — given current technologies and the prevailing knowledge about impact mechanisms and resource levels — are as follows:

- **net CO_2-e emissions**,
- **depletable exergy use**,
- **discharge to air** — as a local impact — noting that such emissions should not be highly localised in their effect, but rather should apply to the entire geographical area of the network under consideration.

The cost function or suite of cost functions chosen for each analysis needs to be established on a case-by-case basis. Furthermore, plant with highly localised impacts should be assessed in its own right prior to inclusion.

3.5 The New Zealand context

This section examines the sustainability crisis from a New Zealand perspective, and reviews the current statutory regime regarding environmental impact entitlements.

The issue of CO_2-e emissions receives particular attention because *deeco* may have a unique role to play in establishing emitter baselines and later managing site-wide discharges.

Environmental change in New Zealand has been broadly similar to that experienced by western industrial nations. Notable differences, however, include a shorter time-horizon, less cross-boarder interactions, a lower population density, and fewer high-concentrations of industry. Useful commentaries include: Baines (1989), Glasby (1991), Ministry for the Environment (1997).

Net CO_2-e emissions

The Kyoto Protocol commitment for New Zealand, signed in December 1997 but not yet ratified, is set at 0.0%. This means that the average net CO_2-e emissions rate over the qualifying years 2008–2012 (inclusive) should not exceed that which occurred during the benchmark year of 1990. The term 'CO_2-e' includes the six gases defined under the UN Climate Convention and listed on p 174. The term 'net' indicates that, in certain circumstances, new forest planting and Third World efficiency projects[7] can be used to offset actual national emissions.

The New Zealand government recently announced its intention to ratify the Kyoto Protocol by mid-2002.

New Zealand is unusual in international terms in two respects. First, the principal net CO_2-e contribution is due to methane, primarily from ruminant farm animals, and not from carbon dioxide from exergy-services processes including transport, as shown in table 3.2. Second, and somewhat dependent on the prevailing hydrology,[8] around 75% of New Zealand's electricity is generated using large hydro, which produces no direct net CO_2-e.[9] The marginal supply at most of New Zealand's main load centres (excluding the Comalco aluminium smelter) is predominantly gas-fired thermal generation, and the exergy sector has shown strong emissions growth.

[7] Known as 'joint implementation' (JI).

[8] That is, rainfall, snow-melt, and evaporation patterns.

[9] For example, hydro comprised 79% of total generation in 1996.

[10] Source: Dang (1999), Ministry of Commerce (1999)

Sector	**Emission rate** [Mt CO_2-e/year]	**% of total**
Energy sector	26	14
Remaining sectors	164	86
National total	190	

Table 3.2 *New Zealand CO_2-e emissions for 1998.*[10]

COP-5 (Conference 5 for the UN Climate Convention, held in Bonn, Germany, November 1999) reporting shows that New Zealand's emissions have risen by 30% over its 1990 benchmark — this being the greatest increase of any Kyoto signatory (Ministry of Commerce 1999). New Zealand has yet to tackle the issue of stabilising its emissions despite 5 years of public policy discussion.

Consultation undertaken by the previous government centred on a national emissions permit trading scheme that could interface later with offshore permits markets, together with a levy system for small emitters (Ministry for the Environment 1999, WOGOCOP 1996). An earlier voluntary reduction scheme for industry (Ministry of Commerce 1994) is widely regarded as having been ineffective. Permit trading schemes are relatively new instruments, although New Zealand has some experience through its marine fisheries quota management system (QMS).[11]

The New Zealand government has not provided, as yet, a credible policy response for slowing net CO_2-e emissions growth, but it seems likely that such emissions will be priced in some form after the next general election scheduled for late-2002.

Environmental impact entitlement

New Zealand's principal environmental legislation is the Resource Management Act 1991 (RMA).[12] Under this regime, legal entitlement to adversely impact the environment is granted by way of either generic planning compliance or via a dedicated resource consent. The process of establishing local planning rules and issuing consents takes place first at the local political level, with subsequent recourse to specialist courts. Unlike the USA, New Zealand has no conception of environmental crime.

Several reviews of the resource and environmental management process in New Zealand have been conducted recently — these include: Ministry for the Environment (1997, ch 4), Parliamentary Commissioner for the Environment (1998). There are claims that the RMA process has concentrated to heavily on amenity value and failed to tackle the wider issue of sustainability in any meaningful way.

[11] Under the QMS regime, operators hold freely tradable quota which allocate the holder a proportion of the total allowable catch (TAC) for the named species. TACs can be revised annually using harvesting models, and may be set at zero to close a particular fishery.

[12] Available at web URL: `http://www.govt.nz`.

Exergy projects typically require resource consents for land-use, water-take, water-release, and contaminant discharge (including heat) to air and water, as appropriate. Arguments relating to impacts of a global nature including net CO_2-e emissions have generally been difficult to sustain at RMA consent hearings. The notable exception was the 1994 400 MW_e Stratford CCGT proceedings, recorded as Williams, Elms, and Johnston (1995), which ruled that the applicant should mitigate carbon dioxide through forestry additionality.[13] The later 120 MW_e Southdown co-generation plant consent hearing ruled matters concerning net CO_2-e discharge out of order, in the absence of a national policy statement on the subject.

Exergy resource depletion

Exergy resource depletion in New Zealand is a complex issue and largely beyond the scope of this thesis. Suffice to say that the Maui natural gas field is due to end production around 2010. The Maui field currently supplies some 25% of New Zealand's primary $nergy.[14]

3.6 Closure

The climate crisis is taken to be the principal biophysical sustainability issue as regards exergy-services supply systems, although the sustainability problem is clearly multi-dimensional. □

[13] The government, through Environment Minister Upton, later diluted this decision (Upton 1995, Ministry for the Environment 1997, box 5.11, p 5.50).

[14] Calculated from Dang (1999).

Chapter 4

Energy modelling and related topics

This chapter reviews the discipline of energy modelling, primarily to locate deeco within this wider field.

4.1 Production systems from a physics perspective

Aspects of contemporary production and consumption systems can be interpreted using concepts derived from physics. These insights can, in turn, be used to inform public policy. One important issue, in context of this thesis, is to better understand the notion of exergy as an input into industrial societies.

Exergy, matter, and information

Exergy, matter, and information are fundamentally inseparable — in the sense that none of these constructs can be formulated without reference to the other two. This statement can be supported in a number of ways. Under macroscopic thermodynamics, the energy-representation fundamental relation describes the behaviour of a single-component simple fluid system with volume V and amount of substance N:

$$U = U(S, V, N) \tag{4.1}$$

Equation (4.1) is monotonically increasing and homogeneous of degree one,[1] based on arguments from statistical mechanics. Inspection reveals that amount of substance N is inseparable from system internal energy U. Furthermore, information about the system, such as its temperature T and pressure P, is similarly based on a knowledge

[1] 'Homogeneous of degree one' requires: $f(\lambda x, \lambda y) = \lambda^k f(x, y)$, with $k = 1$.

of system energy:[2]

$$T = \left.\frac{\partial U}{\partial S}\right|_{V,N} \qquad P = \left.\frac{\partial U}{\partial V}\right|_{S,N} \tag{4.2}$$

The inseparability view can also be supported using arguments from statistical mechanics, information theory, and quantum theory. Although beyond the scope of this thesis, definitions for entropy arising from statistical mechanics and information theory are equivalent (Coveney 1988, Jaynes 1979, Tribus 1979).

From a microeconomics perspective, it is generally considered legitimate to classify a resource flow by virtue of its principal worth to production or consumption — that is as an exergy source, as a material input, or as beneficial information. Thus, microeconomic production functions which treat exergy, matter, and information as individual variables are generally acceptable, but this framework requires an explicit identification of *purpose* for each resource flow input, *eg*, a given mass flow would need to be classified either as an exergy carrier, a feed-stock input, or a signal carrier. Those resource flows which are attributed value on the basis of their exergy content form the focus of this thesis.

Production and consumption chains can be interpreted as out-tree-like (*ie*, sequential) thermodynamic processes, undertaken in order to provide some desired service. The point of interface between production and consumption is determined by market dynamics and not necessarily by physical system considerations. However, industrial production can be viewed as the generalised reverse of mixing — be it thermomechanical (ΔT, ΔP), chemical ($\Delta\mu_i$), and so forth. (Ayres 1994).

Complexity

Far-from-equilibrium (FFE) thermodynamics endeavours to explain the emergence of structure from within dissipative systems. In this respect, industrial society can be interpreted as a dissipative structure, requiring a constant flux of high quality exergy (*ie*, with Z close to unity), most of which ultimately derives from the sun (Prigogine and Stengers 1984). The newer field of emergence theory also tackles this general problem (Bak 1996), as do earlier efforts to integrate themes from thermodynamics, biophysical evolution, and industrial progress (Ayres 1994, Jantsch 1980, Weber, Depew, and Smith 1988). Some workers have attempted to combine equilibrium concepts from economics with those from ecosystems theory and macroscopic thermodynamics (Ruth 1993).

This thesis does not explicitly embrace any of the research fields just reviewed, primarily because it concentrates on a problem of a more operational nature. But

[2] More generally, each extensive variable, such as S and V, naturally pairs with a conjugate intensive variable, in this case T and P, respectively.

equally, neither does the material presented here appear to be at odds with any of the research fields just outlined.

Flow classification in general

The classification of a particular flow is made by examining its primary purpose, noting that no one flow type can exist in isolation:

- exergy flow — using heat, work, and mass transfer,
- signal — dedicated and non-dedicated data-flow channels,
- material flow — for materials processing,
- flow of funds — used in economic modelling.

A similar classification is used in control engineering block models whereby block interconnections can carry information in any of the above forms (Seborg, Edgar, and Mellichamp 1989).

The model given is framed primarily in terms of exergy flows and their flow attributes, although dedicated data-flow channels — which represent control system circuits — may also be included.

Logification

It is currently fashionable to regard western nations as transitioning from commodity-based economies to information-based economies. A better interpretation may be that these economies are simply adopting a more logical view (in the computer science sense) of their underlying physical processes — a development which clearly requires a greater emphasis on the collection, processing, protection, and trading of information, and which is in turn facilitated by greater mechanisation and automation. The process of ′logification′ means that systems can be integrated across a greater scope than was previously possible — a strategy which, depending on the architecture and protocols that prevail, may either assist or exacerbate the containment of failure events.

The *deeco* model, in a sense, exemplifies the trend toward logification. The model minimises some chosen proxy for sustainability in the face of service demand and is framed entirely in terms of data-flow. But the underlying system is clearly reliant on real industrial and biophysical processes. And ultimately it is the biosphere which supplies the required exergy and materials, which receives the waste products, and which carries the impacts.

Arguably, the key determinant of sustainability is not the degree to which logification can and will occur, but the extent to which the overarching system goals specifically support or ignore improved sustainability as an explicit outcome.

4.2 Energy modelling

As discussed, $nergy represents a major factor input for all industrial societies. So not surprisingly, governments, trade-blocks, and corporations put substantial effort into forecasting demand, supply, and price trajectories, in order to help formulate public policy and private strategy. Of late, western governments have sought to influence or regulate $nergy infrastructure investment and management, $nergy markets, and $nergy consumption behaviour, rather than remain directly involved in the provision of $nergy supply.

But the situation is highly complicated — influenced by and influencing: economic activity, social values, technological development, institutional capacity, resource inventory, and environmental status. No meta-model is possible, but aspects of particular $nergy systems of interest can be modelled with some validity.

The field of energy modelling is taken to be the building of numerical models for some $nergy system of interest which attempts to capture and elucidate those system features of importance to the modeller. As such, the modelling strategy and the discipline in which it resides is profoundly linked to the purpose of the model (Bossel 1994). Even so, relatively few energy models are documented comprehensively and published in the public domain — a situation which makes the field difficult to review.[3]

Public sector energy models are generally built to address economic and physical questions. Such models are often used ′comparatively′, meaning that they test input sensitivities rather than produce categorical results. Relatively few energy models explicitly embed the network structure of the systems with which they deal, thus making *deeco* somewhat unusual.

4.2.1 Accounting strategies

Energy modelling can be undertaken using one of two measures of *thermodynamic* account: flow exergy or flow ϵnergy.[4] As discussed in section 5.7 (see p 71), these two measures are generally applied to commodity flows in ′truncated′ form in order to exclude those components in a given resource flow which contain exergy but carry no value in an economic sense. Under this regime, each commodity flow requires an implied (as is often the case) or explicitly stated conversion path. And furthermore, any flow exergy content not captured by the conversion path in mind is simply ′discarded′.

[3] The developers of *deeco* intend to avoid this shortcoming by publishing good documentation as the project unfolds.

[4] Emergy (see p 74) has also been proposed as a measure of account for public policy analysis. Notwithstanding, under neoclassical economics, emergetic content would be regarded as meaningless in terms of consumption decisions.

The most common accounting choice is a relatively unsystematic implementation of truncated flow ϵnergy. The ambient dead-state is often replaced by some laboratory datum, as evidenced by the use of LHV — in this case, liquid water and predefined T_0 and P_0. In practice, further methodological problems exist concerning the points chosen to meter primary $nergy.[5]

The truncated flow ϵnergy regime is widely used to compile national $nergy statistics. The use of ϵnergy is acceptable given that most primary $nergy fuels exhibit an exergy quality factor Z close to unity. But the reporting of geothermal $nergy in this manner (as happens in New Zealand) is quite spurious.

This thesis argues that a more systematic implementation of truncated flow exergy is indicated for $nergy reporting. This measure better captures commodity flows exhibiting below unity exergy quality factors, and more explicitly captures ambient conditions. It may be desirable to factor out some environmental variability (*eg*, by filtering high frequency components) as long as this practice is documented.

Zarnikau, Guermouches, and Schmidt (1996) provide a useful commentary on $nergy statistics, the valuation of $nergy, the and validity of aggregation. van Gool (1987) reviews valuation issues.

4.2.2 Energy modelling for public policy

Numerical public policy energy modelling came of age in the late-1970s in response to the 1974 oil crisis and with the advent of high-level programming environments. At that time, western governments sought to reduce exposure to the economic, social, and political dislocations that occur when the price of crude oil jumps abruptly or the rate of supply constrains. Various models were developed to investigate potential government interventions aimed at improving what was then known as ′energy security′ by stimulating supply diversity and end-use efficiency.

More recently, public policy energy models have been developed and used to investigate $nergy sector deregulation and deregulation strategies — as the same western governments replaced national $nergy planning with investor-driven development. Large corporations also build in-house models to investigate $nergy sector dynamics for business strategy purposes.

With growing acceptance of both the risk and actuality of anthropogenic climate change, energy models, which factor in greenhouse gas emissions, are now being modified or developed anew. The 1997 Kyoto Protocol set specific national net CO_2-e

[5] For instance, under New Zealand's current reporting regime, hydro-electricity is deemed to arise as ′primary $nergy′ at the switchyard. This means that the exergy destruction arising in the inflow between the maximum and actual hydro-reservoir levels is not included, and neither is that within the penstocks, turbines, and generators. At better scheme would be to meter the entitled input as inflow exergy rather than as dispatched electricity.

emissions commitments for the industrialised countries. In the US, policy modellers are attempting to quantify the impact, if any, on economic growth, arising from various proposed emissions restraint interventions.

The energy models in use are methodologically diverse. Bunn and Larsen (1997a) review many of the best known working models. Traditionally, these models were econometric or optimisation-based — but more recently system dynamics models have been developed. These models can be classified as follows (Bunn and Larsen 1997b):

- electricity industry planning models,
- multi-sector optimisation models with bounded infrastructure,
- $nergy/economy interaction-based macroeconomic models,
- economic input/output models,
- computable general equilibrium (CGE) models,
- system dynamics models.

Recent development efforts include the intermeshing of models, including the use of econometric model output to inform physical optimisation. The models listed above have been applied to the following types of public policy issues (Bunn and Larsen 1997b):

- $nergy/economy interactions and economic growth,
- $nergy conservation opportunities,
- natural resource inventories,
- environmental issues including greenhouse gas emissions,
- regulation policy and competition side-effects.

Each class of model has its own particular strengths and shortcomings — some of the key issues are listed below without discussion:

- model scope *versus* model resolution in time and space,
- reliance on exogenous inputs — such as forecast GDP,
- model calibration — particularly for system dynamic models,
- inability to represent dynamic effects — including positive feedbacks,
- the need to assume comprehensive market clearing — that is, no disequilibrium across markets,
- data availability and data management issues,
- algorithm stability and coding quality.

4.2.3 Review of some major models

This section reviews some of the major energy models currently in use, as indicated in table 4.1. However, the topic is complex and the treatment given here is necessarily brief.

Useful reviews of the field include: Bhattacharyya (1996), Kydes, Shaw, and McDonald (1995), Parikh (1998), Ziemba, Schwartz, and Koenigsberg (1980).

Models incorporating sustainability issues, primarily CO_2, are discussed by: Chung, Wu, and Fuller (1997), De Vries *et al* (1994), Lehtila and Pirila (1996), Matsuhashi and Ishitani (1995).

4.3 Optimisation models which embed infrastructure capacity limitations

MARKAL, MESSAGE, EFOM, and MODEST are all optimisation models which embed infrastructure capacity limitations. Some contain econometric modules to project demand and others support net CO_2-e emissions rates as a pre-defined constraint. Two are singled out for discussion because of their similarities with *deeco*: EFOM and MODEST.

4.3.1 EFOM : Energy Flow Optimisation Model

EFOM was first written in the late-1970s but has undergone substantial development. Like *deeco*, EFOM represents an ESSS as a physical network with capacity limits on conversion and transport, specifies exogenous demand in terms of either exergy-service or consumer $nergy, and requires that all demand be met. EFOM also supports high exergy content materials processing (*eg*, cement, steel, and aluminium) as a form of exergy demand. But EFOM optimises for discounted cost and relegates net CO_2-e emissions and/or other environmental costs, if included, as upper bound constraints. *deeco* by contrast adopts the reverse strategy, with monetary cost reported as decision information and not as an explicit model constraint.

The principal difference between the two, however, is the level of spatial and temporal resolution . EFOM uses a highly disaggregated representation of the network and time-intervals are typically in yearly steps. Clearly, EFOM and *deeco* fulfil different roles and each captures different aspects of a particular problem. (Lehtila and Pirila 1996, Bunn and Larsen 1997a, ch 7).

[5] Sources: AEPSOM (Bunn and Larsen 1997a, ch 6), EFOM (Bunn and Larsen 1997a, ch 7, Lehtila and Pirila 1996), IDEAS (Bunn and Larsen 1997a, ch 3), MARKAL (Bunn and Larsen 1997a, ch 5), MESSAGE (Messner, Golodnikov, and Gritsevskii 1996), MIS, (Bunn and Larsen 1997a, ch 4)), MODEST (Henning 1997, 1998), and NEMS (Bunn and Larsen 1997a, ch 2, Koomey *et al* 1998).

Name	Full title	Support	Comment
AEPSOM	Australian Energy Planning System Optimization Model	Australian government	Multi-level optimisation partial equilibrium energy sector microeconomic planning model.
EFOM	Energy Flow Optimization Model	European Union	Aggregated network-based optimisation model seeking minimum discounted cost. Environmental issues can be included as constraints.
IDEAS	Integrated Dynamic Energy Analysis Simulation		System dynamics supply/demand model which caters for time lag between demand and supply. Used to model greenhouse gas policy.
MARKAL	MARKet ALlocation	International Energy Agency	Dynamic linear programming multi-period model of the technical energy system. Can include CO_2 constraints. MARKAL-MACRO has a coupled economic model which drives the main energy optimisation routine. Currently undergoing major rewrite using GAMS.
MESSAGE	Model for Energy Supply Systems Analysis and their General Environmental impact	International Institute for Applied Systems Analysis	Environmentally Compatible Energy Strategies (ECS) Project
MIS	Macroeconomic Information System	German government	Dynamic input/output model. Applied to carbon tax evaluation.
MODEST	Model for Optimization of Dynamic Energy Systems with Time-dependent components and boundary conditions	Linköping Institute of Technology, Sweden	LP-based optimisation model with support for storage. Demand represented by peak and average for each interval. Suitable for municipal problems.
NEMS	National Energy Modeling System	Energy Information Administration, US Department of Energy	Modular sectorial model. Used in LBL study.

Table 4.1 *Major energy models.*[5]

4.3.2 MODEST : Model for Optimization of Dynamic Energy Systems with Time-dependent components and boundary conditions

MODEST, like *deeco*, is an LP-based energy system decision support tool designed to evaluate fuel type, technology, storage, and load management options. The model has been applied to several municipal and national studies, mostly in Scandinavia. MODEST reports supply curves, broken down in terms of actual or displaced fuel type, which are informative. MODEST claims to be able to better optimise storage than MARKAL, MESSAGE, and EFOM. The principal differences with *deeco* are lower temporal resolution and hence loss of important demand/supply correlations, less sophisticated process modelling, and the inability to optimise for environmental cost. (Henning 1997, 1998).

4.4 Energy modelling in New Zealand

At the time of writing, New Zealand does not undertake any form of integrated national-level energy and/or CO_2 policy modelling, beyond producing supply and demand projections. Instead, reliance has been placed on the $nergy sector reform programme, this being driven until recently by *laissez-faire* economic philosophies. The reforms were obstensively based on argument about improved, but non-internalised, allocative efficiency and consequent economic growth. Arguably, the $nergy industry sanctioned these changes on the understanding that they would retain the revaluation windfall, whilst major users expected to benefit from the removal of domestic cross-subsidies and from price discrimination opportunities.

4.4.1 Models and methods presumed no longer in use

The older models are no longer directly relevant, as the sector and the policy problems which they addressed have changed markedly. Nonetheless this work is listed as it may still contain useful information.

Smith (1978) describes an LP-optimation model of the New Zealand $nergy supply and distribution system developed at Victoria University. Kask (1988), Pearson (1977), Peet and Baines (1986) review $nergy analysis methodologies. Hendtlass, Peet, and Baines (1988) report an $nergy scenario input/output model which contained an econometric module to furnish demand projections.

Patterson (1987) reviews issues related to systems appraisal, exergy quality, and process efficiency. More recently, Patterson (1993) reviews second law efficiency formulations. Patterson proposes the following index as the candidate that best meets certain considerations, although any theoretical restrictions are not identified (p 23) (the notation is specific to Patterson):

$$E_{\Delta G} = \frac{\Delta G_{\text{out}}}{\Delta H_{\text{in}}} \tag{4.3}$$

where: $E_{\Delta G}$ is the proposed measure of second-law efficiency, ΔG_{out} represents process outputs measured in terms of their Gibbs free energy, and ΔH_{in} represents process inputs measured in terms of their enthalpy, both relative to ambient. This expression should be compared with eqn (5.17) (see p 65).

Wholesale electricity market simulations were written in the early-1990s. Energy Group Ltd, Dunedin, and physicist Dr Alan McCord undertook some of this work.

4.4.2 Contemporary modelling efforts

Energy Research Modelling Group

The University of Canterbury Energy Research Modelling Group (EMRG) is actively tackling electricity sector problems from an operational research perspective. EMRG have recently undertaken optimisation modelling for hydro-storage (Scott and Read 1995, Yang and Read 1999) and wholesale electricity pricing issues including: nodal pricing (Hogan, Ring, and Read 1995), transmission pricing (Read 1997), and reserve pricing (Read, Drayton-Bright, and Ring 1998).

EERA*: Energy Efficiency Resource Assessment*

Rossouw, Lermit, and James (1997) report progress on a consumer $nergy demand forecast model which projects end-user exergy-services demand, technological progress, and demographic trends, and sums these contributions. The BAU baseline, once established, can be used to quantify consumer $nergy savings potentials arising from accelerated end-use efficiency uptake. These results can then be used to inform public and/or private policy development. The model is written in VBA as an Microsoft Access database application.

Wholesale electricity nodal pricing

The New Zealand electricity sector has developed a sophisticated nonlinear programming (NP) model[6] designed to establish *ex-ante* (*ie*, forecast) nodal pricing[7] and to prioritise generator dispatch — the software is called 'SPD' (scheduling, pricing, and dispatch). Like *deeco*, SPD contains a capacitated flow network representation and seeks minimum flow-weighted aggregate cost — in this case the objective is the monetary cost faced by demanders. But unlike *deeco*, SPD undertakes market clearing at each node in order to allocate available supply and strike a series of

[6] Read (1998, p 9) reports that the underlying model is piecewise linear.

[7] At the time of writing, final pricing is still being advised after the fact.

nodal prices. SPD itself is operated in two roles. Initially, M-co,[8] who administers the New Zealand Electricity Market (NZEM), runs the model in 30 min blocks to set the *ex-ante* price based on generator bids and load projections. Subsequently Transpower, the national grid operator, runs the model in real-time to determine the price-optimal generator dispatch regime in light of actual demand.

In addition, the major generators and various third-party consultants have developed similar in-house models for analysis and forecasting purposes. For example, the Energy Link product is called ′Emarket′.[9]

Embedded wind-generation optimisation study

Industrial Research (IRL) [10] have investigated the optimal placement of wind-turbines in weak distribution networks, primarily to defer or avoid lines upgrades. The study is two-fold: first nonlinear programming (NP) techniques are used to screen potential turbine sites using economic criteria, and second, an electrical system simulation is used to quantify the benefits in terms of voltage support and power factor correction. The power electronics in state-of-the-art wind turbines can be used for power conditioning when not generating, and this facility may be of considerable value. (Ackermann, Garner, and Gardiner 1999).

Ministry of Commerce scenarios

The Ministry of Commerce's Energy Modelling and Statistics Unit produce regular energy outlooks, using a partial equilibrium $nergy sector model called SADEM — the latest release being Ministry of Commerce (2000). The assumptions used in these projections are not without controversy — in particular, future oil price estimates and efficiency uptake projections both attract criticism.

4.5 Site-wide thermoeconomic optimisation

Two thermoeconomic optimisation-related techniques warrant inclusion: pinch analysis and cost-optimal design analysis. Pinch analysis is accorded some depth because it may overlap with *deeco* in terms of application, particularly for site-wide emissions management.

4.5.1 Pinch analysis

Pinch analysis, in its simplest form, is an SSSF exergy-based optimisation heuristic used for the site-wide integration of heat-exchanger and CHP networks, with the

[8] Legally registered as: The Marketplace Company Ltd.

[9] The web URL is: `http://www.energylink.co.nz`.

[10] IRL is a state-owned commercially run research institute.

goal of reduced capital and fuel costs.[11] Pinch analysis is summarised in figure 4.1. The starting point for the method is to identify all the heat transfer needs for the site and then seek a heat exchange configuration that leads to the best identifiable use of heat and cooling utilities by reducing unnecessary exergy destruction. The designer can trade plant costs *versus* exergy costs, by varying the least heat exchange temperature differential (ΔT_{min}) and, in addition, can look for opportunities to re-specify some processes to advantage. The next step is to examine potential design rationalisations. The designer can now trade heat recovery *versus* component count, by splitting and/or combining the transfer flows. In addition, heat-engines, heat-pumps, refrigeration units, and expanders can be included using certain rules and similarly evaluated. In essence, the method concentrates on system integration, rather than component performance. Pressure drop exergy destruction is assumed secondary in magnitude to heat transfer exergy destruction. Practitioners claim operability is not necessary compromised as a result of applying pinch analysis.

The method has been extended since first published in 1979, but following commercialisation, recent developments are no longer reported in detail in the public domain. These developments include: batch processing, waste and waste-water minimisation, emissions targeting, and total site integration. The core method, as described above, is covered in Linnhoff *et al* (1982) and Bejan, Tsatsaronis, and Moran (1996, p 473–498), whilst Linnhoff (1993) reviews 15 years of development. Pinch analysis has been applied to a New Zealand fertiliser works to advantage (Ministry of Commerce 1992).

4.5.2 Cost-optimal design analysis

Thermoeconomic optimisation covers the analysis, design, and integration of thermal systems for best financial benefit, a task also known as 'cost-optimal' (CO) design. The optimisation techniques employed form a research area in that these are yet to move from analysis-informed design to explicit mathematical optimisation. The central tenet is that the design of a component can be interpreted as a compromise between exergetic destruction and loss *and* amortised financial cost. For example, the following component-based ratio can be calculated for component k (Bejan *et al* 1996, eqn 8.28, p 438):

$$f_k = \frac{\dot{Z}_k}{\dot{Z}_k + c_{F,k}(\dot{E}_{D,k} + \dot{E}_{L,k})} \tag{4.4}$$

where: f_k is the exergoeconomic factor for component k [–]
$\dot{Z}_k$ is the non-exergy related levelised financial cost rate [\$/s]
$c_{F,k}$ is the cost of fuel [\$/kJ]

[11] A heuristic is an algorithm which gives an answer which is good but not necessary optimal (Sedgewick 1983).

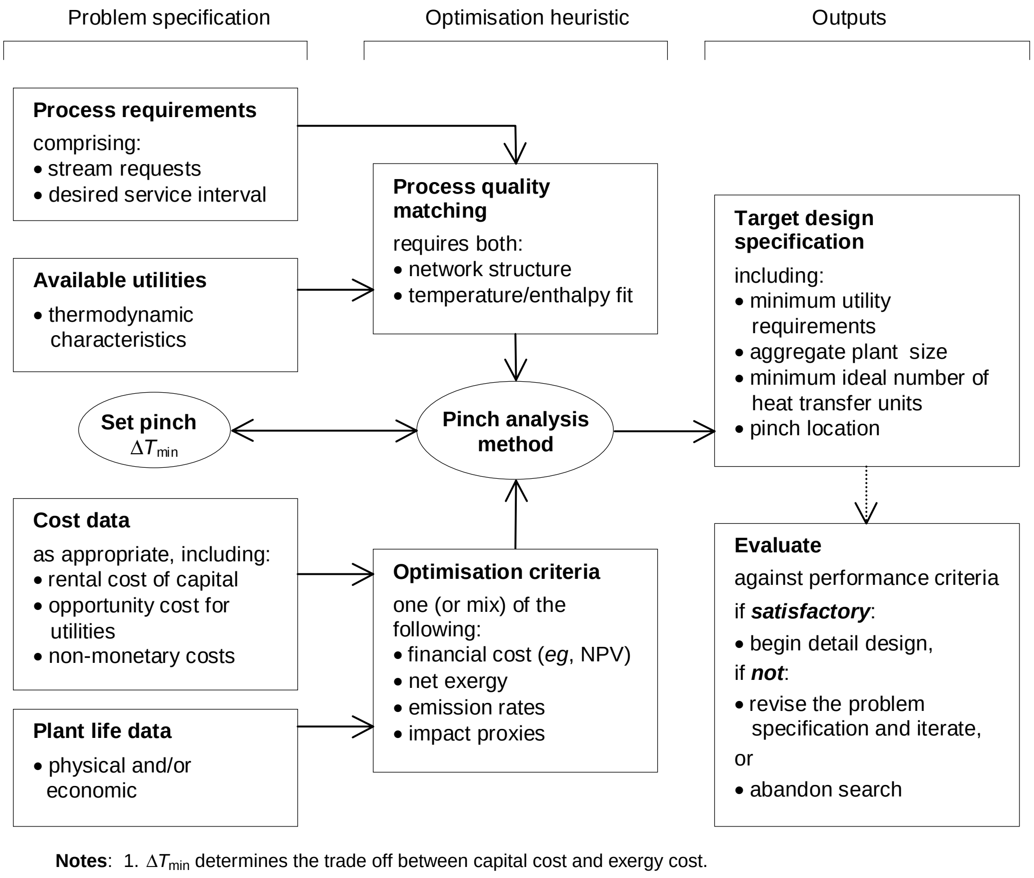

Problem specification
Optimisation heuristic
Outputs
Process requirements
comprising:
• stream requests
• desired service interval
Available utilities
• thermodynamic characteristics
Set pinch
$\Delta T_{\min}$
Cost data
as appropriate, including:
• rental cost of capital
• opportunity cost for utilities
• non-monetary costs
Plant life data
• physical and/or economic
Process quality matching
requires both:
• network structure
• temperature/enthalpy fit
Pinch analysis method
Optimisation criteria
one (or mix) of the following:
• financial cost (eg, NPV)
• net exergy
• emission rates
• impact proxies
Target design specification
including:
• minimum utility requirements
• aggregate plant size
• minimum ideal number of heat transfer units
• pinch location
Evaluate
against performance criteria
if satisfactory:
• begin detail design,
if not:
• revise the problem specification and iterate,
or
• abandon search
Notes: 1. $\Delta T_{\min}$ determines the trade off between capital cost and exergy cost.

Figure 4.1 *An overview of pinch analysis.*

$\dot{E}_{D,k}$ is the exergy destruction rate [kW]
$\dot{E}_{L,k}$ is the exergy loss rate [kW]

The non-exergy-related levelised financial cost rate requires the selection of both an economic life for the plant and a real discount rate. Comparing exergoeconomic factors for components of the same class (*eg*, turbines) can give a clue to the quality of the component and comparing factors for components within a particular system can help identify development priorities. This and similar techniques are reviewed by Bejan *et al* (1996) and Bejan and Mamut (1999).

4.6 Closure

There has been a resurgence of interest in national-level energy modelling internationally. The reasons are twofold. First, the recent adoption of national net CO_2-e emissions quota under the 1997 Kyoto Protocol means that governments are now seeking to incentivise $nergy sector emissions reductions, stimulate development in low-carbon technologies, influence private behaviour, and include carbon-related impacts in national accounting as far as is deemed reasonable. Second, $nergy sector liberalisation has stimulated econometric-based modelling designed to gain insight into the dynamics which can arise under the new regimes.

deeco is not intended to compete with the large-scale dynamic policy models reviewed in this chapter — rather *deeco* is primarily a decision support tool which can assist both public and private sector managers make environmentally beneficial decisions in terms of the constraints envelope they face. In addition, *deeco* can be used to benchmark $nergy sector practice or evaluate publicy policy inteventions — it is in this latter context that the application of *deeco* to a national policy model is discussed in chapter 9. □

Chapter 5

Exergy processes and exergy plant

This chapter discusses exergy processes, their approximation as multi-connection plant, and the adaptation of these and related ideas for use in a flow network model. These ideas underpin deeco.

5.1 Introduction

This chapter describes certain process concepts integral to the network model under discussion — these include context-dependent thermodynamic functions, the bulk commodity approximations, the multi-connection plant abstraction, the ideas of logical exergy flow and logical plant, and the flow network model abstractions of sourcing plant and demand plant. The exergy functions contained within this chapter are presented with more formalism in appendix B. The treatment adopted in this thesis is based on Bejan (1997).

Readers may wish to review appendix A which defines key teminology before continuing with this chapter.

5.2 Thermodynamic functions

This section reviews established thermodynamic functions needed to quantify the model being pursued, and introduces two new functions: flow ϵnergy and nonflow ϵnergy.

State and context-dependent functions

Thermodynamic functions are used to describe thermodynamic systems, and often identify some salient extensive property in terms of measurable variables. Extensive thermodynamic functions apply to all systems, but only uniform systems can be assigned one set of intensive variables.[1] The cut-point and the bulk-storage approximations (see p 60), which require uniformity, allow flow and stock commodities to be conveniently assessed.

The thermodynamic functions used in this thesis fall into two types: state functions *and* context-dependent functions. The latter type are important because they better capture the notion of value as applied to commodity assessment, and can be used to undertake plant performance evaluation. The context-dependent functions include both state functions and context variables in their definition.

State functions

State functions are solely dependent on system conditions and are independent of the prevailing environmental conditions. The state functions of interest in this thesis are U, H, S, and $\bar{E}$. Fundamental explanations for these functions can be found in statistical physics — Kittel and Kroemer (1980) give an introductory treatment.

Readers unfamiliar with state functions simply need to know that values of U, H, and S for commonly encountered systems are published in physical chemistry databooks or coded into computer procedures.[2] The values themselves are calculated from measurements obtained from certain informative controlled processes.

Context-dependent functions

The context-dependent functions used in this thesis divide into environment-dependent functions and environment- and market-dependent functions, as shown in table 5.1, As noted previously, the flow and nonflow ϵnergy functions are novel.

5.3 Environment-dependent functions

The environment-dependent functions of exergy and ϵnergy can be used for two purposes: commodity assessment and plant performance evaluation. Of the two functions, exergy is arguably the more intuitive, but ϵnergy often furnishes additional information about the situation under investigation, particularly regarding unaccounted first law losses. Environmental dependence provides both functions

[1] 'Uniform systems' are more properly known as 'simple systems'. 'Uniformity' indicates that the system be macroscopically homogeneous, isotropic, and subject to spatially invariant external fields.

[2] These are normally listed in mass-specific and/or molal forms.

Extensive function type	Name	Symbol
State	Internal energy	U
	Enthalpy	H
	Entropy	S
	Total energy	$\bar{\bar{E}}$
Environment-dependent	Flow exergy	E
	Nonflow exergy	Ξ
	Flow εnergy	L
	Nonflow εnergy	Λ
Environment- and market-dependent	$nergy	Σ

Table 5.1 *Extensive state and context-dependent thermodynamic functions utilised in this thesis.*

with their usefulness, but this dependence also means that measures of commodity inventory or plant performance will, in general, change whenever the environmental conditions vary. Exergy and ϵnergy are strictly technical concepts and unlike $nergy do not require judgements as to the purpose and worth of the commodity flow in order to proceed. The plant performance measures adopted in this thesis do require a statement of plant purpose, but this appraisal need not be economic. Plant performance evaluation requires that exergy or ϵnergy balances, as appropriate, be applied to the plant or collection of plant under analysis.

The two environment-dependent functions of primary interest, exergy and ϵnergy, do not possess fully general statements — rather, the particular formulation will depend on circumstances, with selection issues grouped as follows:

- flow and nonflow commodity assessment,
- thermomechanical (TM) and total formulations (these apply to mass-flow),
- system context.

Readers wishing to examine the mathematical definitions for particular exergy and ϵnergy functions at this point should consult p 57–58.

Flow and nonflow commodity assessment

Commodity assessments naturally separate into flow and nonflow forms covering flow and stock commodities respectively. These two types stem from the balance equations used to establish the operational meanings for exergy and ϵnergy. Both types are required because, in general, an ESSS is NSSF for each time-interval. The inter-plant transactions are quantified using flow-based assessment and the intra-plant storage transactions are quantified using nonflow-based assessment.

Thermomechanical and total formulations

It is convenient to formulate mass-based exergy and ϵnergy in two ways, depending on whether or not the stream or stock commodities have the opportunity to interact chemically with their environment — this chemical interaction need not be reactive and includes species diffusion and transport. The two formulations are known as ′thermomechanical′ and ′total′, and cover containment and interaction, respectively. The differences between these two formulations are summarised in table 5.2. Neither strand is fully encompassing, so the term ′total′ is somewhat misleading.[3]

Attribute	**Form**	
	TM	Total
Kinetic exergy of stream	yes	no
GP exergy of stream	yes	no
Species-permeable boundary	no	yes
Treatment of matter	mass	mole
Subscript	x	t
See figure	B.1	B.2

Table 5.2 *Thermomechanical and total formulation of mass-based exergy and ϵnergy.*

Chemical exergy and ϵnergy are defined as the total exergy and ϵnergy, respectively, of a resource with nil kinetic and/or gravitational potential exergy which is also at T_0 and P_0 — the ′restricted′ dead-state. This same resource can then be brought into chemical equilibrium with $\mu_{0,i}$ in order to recover its chemical exergy and ϵnergy.

Summary

The suite of exergy and ϵnergy functions and formulations which result from the breakdown presented in the previous two sub-sections is summarised in table 5.3.

[3] These two strands could be generalised into one, but in operational terms there would be little point in doing so.

Function	Type	Form	Symbol				Eqns (text)	Eqns (appendix)
			Extensive	Mass-specific	Molal	Non-mass		
Exergy	Flow	TM	E_{x}	e_{x}			5.1	B.20
		Total	E_{t}		$\bar{e}_{\mathrm{t}}$		5.3	B.32
		Heat				E_Q	5.5	B.22
		Work				E_W	5.7	—
	Nonflow	TM	Ξ_{x}	ξ_{x}			5.9	B.17
		Total	Ξ_{t}		$\bar{\xi}_{\mathrm{t}}$		5.11	B.30
ℑnergy	Flow	TM	L_{x}	ℓ_{x}			5.2	—
		Total	L_{t}		$\bar{\ell}_{\mathrm{t}}$		5.4	—
		Heat				L_Q	5.6	—
		Work				L_W	5.8	—
	Nonflow	TM	Λ_{x}	λ_{x}			5.10	—
		Total	Λ_{t}		$\bar{\lambda}_{\mathrm{t}}$		5.12	—

Table 5.3 *Flow and nonflow exergy and ϵnergy— notation and equations.*

Use of subscripts: the exergy and ϵnergy variants are defined using subscripts to indicate their form. These subscripts may be omitted when general statements are being made, but a particular problem will require that the appropriate formulation be used. Put in mathematical terms:

$$\begin{aligned} E &\in \{E_{\mathrm{x}}, E_{\mathrm{t}}, E_Q, E_W\} \\ L &\in \{L_{\mathrm{x}}, L_{\mathrm{t}}, L_Q, L_W\} \\ \Xi &\in \{\Xi_{\mathrm{x}}, \Xi_{\mathrm{t}}\} \\ \Lambda &\in \{\Lambda_{\mathrm{x}}, \Lambda_{\mathrm{t}}\} \end{aligned}$$

In some cases, the $_{\mathrm{x}}$ or $_{\mathrm{t}}$ forms will need to be selected, again depending on circumstances. This situation is indicated using the following notation:

$$\begin{aligned} E_{\mathrm{x}|\mathrm{t}} &\in \{E_{\mathrm{x}}, E_{\mathrm{t}}\} \\ L_{\mathrm{x}|\mathrm{t}} &\in \{L_{\mathrm{x}}, L_{\mathrm{t}}\} \\ \Xi_{\mathrm{x}|\mathrm{t}} &\in \{\Xi_{\mathrm{x}}, \Xi_{\mathrm{t}}\} \\ \Lambda_{\mathrm{x}|\mathrm{t}} &\in \{\Lambda_{\mathrm{x}}, \Lambda_{\mathrm{t}}\} \end{aligned}$$

System context

The exergy and ϵnergy expressions in their various guises are most often formulated for fluid systems subject to uniform gravity. Additional $\dot{m}$ or $\dot{N}$ terms, as appropriate, are needed for other system contexts, for instance if other forms of exergy inventory (*eg*, elastic strain exergy) exist or if external fields other than gravity (*eg*, magnetic fields) are present.

The functions given in table 5.3 are defined *en masse* below, and are presented without derivation. The expressions are based on figures B.1 (p 192) and B.2 (p 195), as appropriate. These equations can be regarded as standard results, but if need be, they can be derived from first principles using information contained in appendix B. The symbols have their usual meanings or are defined immediately following a set of equations or are given in appendix E (see p 219).

> **Dot notation**: the dot notation indicates flow-rate and not the more general meaning of time rate-of-change. This restriction arises because the expressions given should properly be expressed as vector-valued equations using the unit vector parallel to flow, but convention allows these ideas to be given in arithmetic form.

5.3.1 Flow commodity definitions

$$\dot{E}_\mathrm{x} = \dot{m}e_\mathrm{x} = \dot{m}\left((h-h_0)+\frac{(\upsilon^2-\upsilon_0^2)}{2}+g(z-z_0)-T_0(s-s_0)\right) \tag{5.1}$$

$$\dot{L}_\mathrm{x} = \dot{m}\ell_\mathrm{x} = \dot{m}\left((h-h_0)+\frac{(\upsilon^2-\upsilon_0^2)}{2}+g(z-z_0)\right) \tag{5.2}$$

$$\dot{E}_\mathrm{t} = \dot{N}\bar{e}_\mathrm{t} = \dot{N}\left((\bar{h}-T_0\bar{s})-\sum_{i=1}^{n}(\bar{h}_{0,i}-T_0\bar{s}_{0,i})\,x_i\right) \tag{5.3}$$

$$\dot{L}_\mathrm{t} = \dot{N}\bar{\ell}_\mathrm{t} = \dot{N}\left(\bar{h}-\sum_{i=1}^{n}\bar{h}_{0,i}\,x_i\right) \tag{5.4}$$

$$\dot{E}_Q = \dot{Q}\left(1-\frac{T_0}{T}\right) \tag{5.5}$$

$$\dot{L}_Q = \dot{Q} \tag{5.6}$$

$$\dot{E}_W = \dot{W} \tag{5.7}$$

$$\dot{L}_W = \dot{W} \tag{5.8}$$

where: $\dot{E}_\mathrm{x}$ is the thermomechanical flow exergy rate of the stream

$\dot{E}_\mathrm{t}$ is the total flow exergy rate of the stream

$\dot{E}_Q$ is the flow exergy rate associated with heat transfer, excluding thermal radiation[4]

$\dot{E}_W$ is the flow exergy rate associated with work transfer

$\dot{L}_\mathrm{x}$ is the thermomechanical flow ϵnergy rate of the stream

$\dot{L}_{\mathrm{t}}$ is the total flow ϵnergy rate of the stream
$\dot{L}_Q$ is the flow ϵnergy rate associated with heat transfer
$\dot{L}_W$ is the flow ϵnergy rate associated with work transfer
x_i is the mole fraction
i is the species index
$_0$ indicates dead-state value

5.3.2 Nonflow commodity definitions

$$\Xi_{\mathrm{x}} = U - U_0 + P_0(V - V_0) - T_0(S - S_0) + \begin{array}{c}\text{macroscopic}\\\text{exergy inventory}\end{array} \tag{5.9}$$

$$\Lambda_{\mathrm{x}} = U - U_0 + P_0(V - V_0) + \begin{array}{c}\text{macroscopic}\\\text{exergy inventory}\end{array} \tag{5.10}$$

$$\Xi_{\mathrm{t}} = U + P_0 - T_0 S - \sum_{i=1}^{n} (\bar{h}_{0,i} - T_0 \bar{s}_{0,i})\, N_i + \begin{array}{c}\text{macroscopic}\\\text{exergy inventory}\end{array} \tag{5.11}$$

$$\Lambda_{\mathrm{t}} = U + P_0 - \sum_{i=1}^{n} \bar{h}_{0,i}\, N_i + \begin{array}{c}\text{macroscopic}\\\text{exergy inventory}\end{array} \tag{5.12}$$

$$\tag{5.13}$$

where: Ξ_{x} is the thermomechanical nonflow exergy of the control volume
Ξ_{t} is the total nonflow exergy of the control volume
Λ_{x} is the thermomechanical nonflow ϵnergy of the control volume
Λ_{t} is the total nonflow exergy of the control volume

Note also that if the system is at P_0, then eqn (5.10) becomes equivalent to:

$$\Lambda_{\mathrm{x}} = H - H_0 \Bigg|_{P = P_0} + \begin{array}{c}\text{macroscopic}\\\text{exergy inventory}\end{array} \tag{5.14}$$

The term 'macroscopic exergy inventory' is required rather than the more common term 'mechanical energy' because the latter need not use the prevailing dead-state as its datum (Kotas 1995, p37–38).

The next sections discuss exergy and ϵnergy in qualitative terms.

5.3.3 Exergy

Exergy is the potential of a stock or flow resource to produce work, in context of the local environment as defined by the dead-state. Exergy can never be fully recovered as work in some desired form if a conversion process is needed to realise this work

[4] The correct expression for the flow exergy of thermal radiation is under some dispute (see p 66).

or change its form. Exergy is therefore destroyed in real processes, a phenomenon represented by the dissipation term in an exergy balance. This means that although exergy is not conserved, it does remain additive — or, more precisely, exergy is an extensive quantity.

The exergy concept, when applied to resource assessment, is general in the sense that a conversion pathway (such as a fuel cell or pelton wheel) need not be specified, but situation-specific in that all contributions arising from the environmental context must be identified. This latter point includes external fields, but normally only gravity requires consideration. On the other hand, the exergy concept when applied to process performance assessment is process-specific.

Exergy analysis requires a suitable dead-state definition. Closed-circuit systems need only certain physical attributes (*eg*, ambient temperature and perhaps pressure) to be identified. But open-circuit systems which interact chemically, but not necessarily reactively, with their environment also require the identification of a suite of suitable reference compounds. These compounds are not the same as used in analytical chemistry — for example: for elemental carbon, CO_2(g) at atmospheric concentration might be used, rather than graphite C(s).

Exergy is favoured because it combines potential work contributions from both mechanics and thermodynamics (in its strict sense), and because it accounts for the influence of ambient conditions on the usefulness of a resource. But exergy also faces methodological difficulties. The principal concern involves the selection of a suitable dead-state reference, given real environments are not in true thermodynamic equilibrium. This shortcoming can usually be resolved for a specific analysis, but it does tend to limit the usefulness of the method across divergent environments — that is, separated in time or place.[5]

The concept of exergy is constructed by applying energy and entropy balances to a generic open system which also exchanges heat and volumetric work, and, in some cases, mass, with its surroundings, and by invoking the lost-work theorem, eqn (B.1). The concept of exergy therefore includes the second law of thermodynamics.

The top half of table 5.3 indicates the various formulations of exergy. Chemical exergy is not shown, but is defined as that component of total exergy which remains

[5] These methodological difficulties do impact on the *deeco* model but the ′errors′ are minor in comparison with other modelling rationalisations. *deeco* factors in such considerations via environmentally-dependent εnergetic efficiency descriptions (which represent as input-output equation sets), and inter-plant exergy traffic is accounted using εnergy not exergy. The process descriptions include any environmental influences as independent variables on an *ad hoc* basis, which thereby removes the need for an explicit integrated dead-state definition. The preceding comments could indicate that an in-depth coverage of exergy is not warranted in this thesis. However, exergy is the one concept that unifies the description of energy systems and does so in a cogent manner. Furthermore, it would be possible for a future rewrite of *deeco* to be framed in terms of exergetic efficiency and account for inter-plant traffic using exergy based on an integrated dead-state.

when a resource with nil kinetic and potential exergy is also at T_0 and P_0.

The writer has been unable to locate a single comprehensive text on exergy analysis. Notwithstanding, Carrington (1994) and Bejan (1997) dovetail together well to complete the underlying theory, and Kotas (1995) covers application issues in depth. ENERPAC (Nieuwlaar 1996) can be used to undertake numerical exergy analysis and contains support for several common environmental reference systems (ERS). Marcella (1992) and Rosen (1999) provide useful reviews of second law analysis.

5.3.4 ℑnergy

ℑnergy is defined as the potential of a stock or flow resource to produce heat, in context of the local environment as defined by the dead-state. The concept of ϵnergy is constructed by applying an energy balance to the same generic open system used to establish the concept of exergy. ℑnergy can be used to create an ϵnergy balance — which can be regarded as an environmental context-dependent energy balance.[6]

Inspection of eqns (5.2), (5.4), (5.10), and (5.12) shows that when the commodity involves mass flow or mass storage, the ϵnergy functions require a dead-state definition. This situation is no different to that which applies to exergy. In other words, combustion processes should reference any water in the waste stream to the prevailing ambient humidity.[7]

5.4 The bulk commodity approximations

Two related approximations are necessary in order to apply the thermodynamic functions previously discussed — the first covers flow regimes and the other nonflow regimes. Both allow complex situations to be captured using a very much simplified description and both are commonly applied in engineering analysis.

5.4.1 The bulk-flow approximation

A commodity flow of arbitrary type is shown in figure 5.1. The commodity itself remains unspecified, but the discussion here is restricted to commodity flows useful primarily for their flow exergy content — that is, deriving from heat, work, or mass flow. The bulk-flow approximation, if valid, allows complex (and not necessarily fluid state) flow to be adequately characterised by a single set of thermodynamic (including mechanics) variables. The cut-point [8] therefore represents a non-isolated

[6] Energy and ϵnergy share the same relationship as availability (see p 188) and thermomechanical exergy.

[7] The problem of assigning ϵnergy content to an oxidisable fuel is more involved in practice. Nieuwlaar (1996, p 39–42) and Kotas (1995, p 258–259) discuss the issues.

[8] The term 'cut-point' is coined in the absence of a better alternative.

uniform system of cross-sectional area and zero thickness. Cut-points should be located where real measurements can be taken, or failing this, where satisfactory estimates can be established. Bejan (1997, p 70) describes this approximation, but refers to it as the "bulk flow assumption".

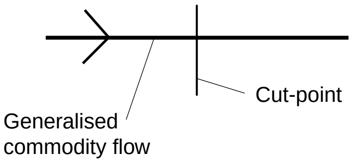


Figure 5.1 *Schematic representation of a commodity flow of arbitrary type and its associated cut-point.*

Because the cut-point system (*ie*, zero-volume control volume) is non-isolated, exergy transfer is only one of several transactions that can be accounted for across the plane of the cut-point — other transactions include: flow ϵnergy, entropy, species-specific mass, linear momentum, and angular momentum (such balances constitute the workhorses of thermodynamic systems analysis).

5.4.2 The bulk-storage approximation

The second manifestation of the bulk commodity approximation is the bulk-storage approximation which, if applicable, allows exergy storage to be adequately characterised by a single set of thermodynamic variables. This approximation includes the bulk storage intensive variables — which are required so that any exergy flows which arise are in turn fully characterised. For instance, thermal hot water storage would need to be approximately free from stratification, or alternatively, modelled as several connected storage systems.

5.5 Multi-connection plant and related commodity flows

5.5.1 Generalised multi-connection NSSF plant

A generalised multi-connection NSSF plant using actual rather than logical exergy flows (see p 71), is depicted in figure 5.2, noting that some of the ideas contained in this figure are given as definitions in appendix A. The figure itself is based on figure B.2 (see p 195), but with the mass flows quantified using $\dot{m}$ rather than $\dot{N}$. In general, plant may transact extra-flow mass, heat, and volumetric work directly with their surroundings, in addition to the exergy flows which cross the cut-points. ′Extra-flow′ transactions are those which do *not* take place across a cut-point, but rather occur directly with the environment (see also p 170). Due to the placement of the system boundary, all exergy destruction related to the plant in question is deemed to take place within the volume that this boundary entrains. Therefore, the boundary will invariably be beyond the surface of the outer wall of the plant in question (refer to Kotas 1995, ch 4). Multiple mechanism transfers, such as stream-wise heat transfer, are excluded from this treatment — including this kind of mechanism would add a layer of complexity not warranted in the model being pursued.[9]

5.5.2 Plant performance measures

Plant performance indexes, irrespective of type, require an assessment of purpose. The following generalised classification is useful when defining such measures:

- **fuel** includes any heat, work, or mass flow or stock exergy input which exhibits economic opportunity cost,
- **product** includes any exergy resource or exergy-service which has economic value,
- **waste** includes any unavoidable or unintentional process — if later used as a fuel or product in its own right, it should be then classed as a co-product.

When undertaking a performance assessment, all control regions need to be explicitly defined and all transactions identified. The classifying of ′product′ and ′fuel′ flows may not be obvious in certain cases and judgement may be needed.

Plant performance measures may be expressed in two forms:

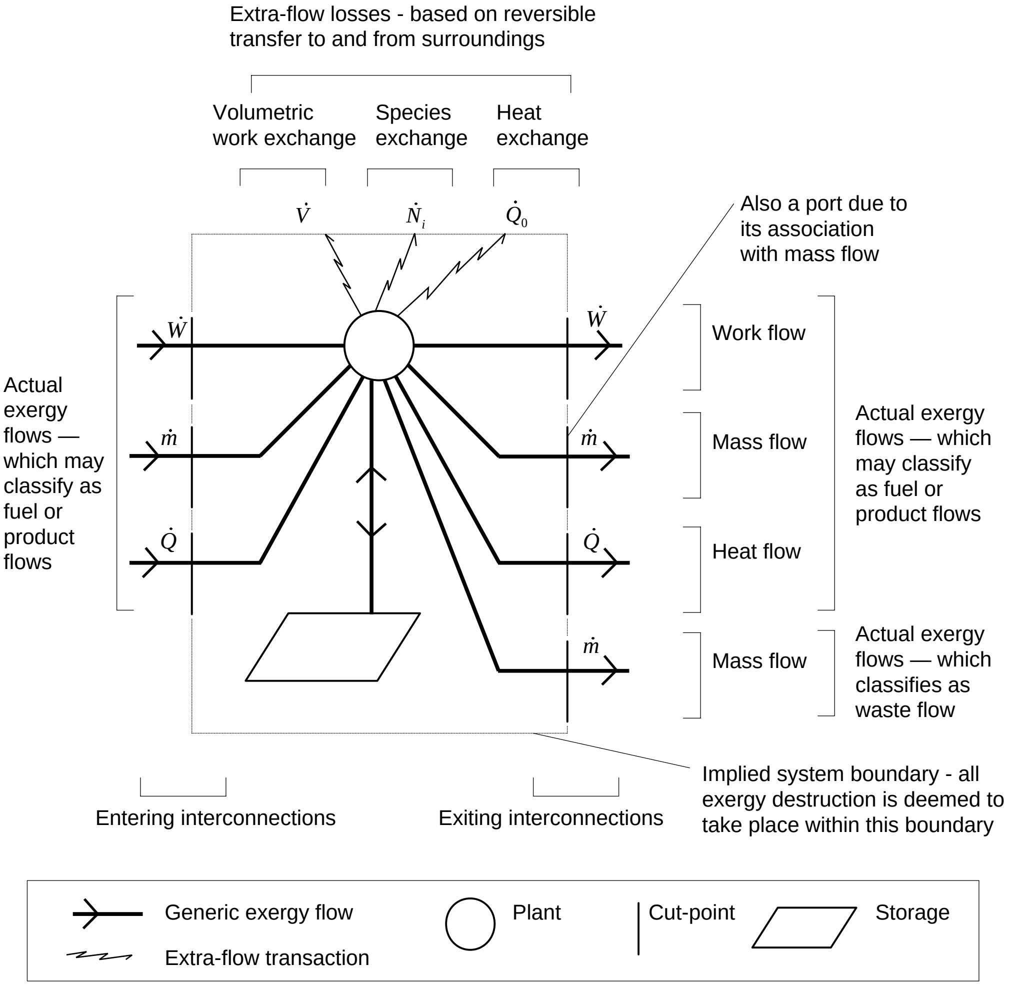
- average measure,
- marginal measure.

More completely, plant performance should be given as an explicit function of the performance-dependent variables.

Second law-based SSSF plant efficiency

Exergy process performance can be assessed using a number of performance indexes. Second law-based measures which compare real and ideal performance, often provide insights not available using first law based measures. The second law-based index

[9] Sun and Carrington (1991) review the treatment of such mechanisms in complex flow.


Extra-flow losses - based on reversible transfer to and from surroundings
Volumetric work exchange
Species exchange
Heat exchange
$\dot{V}$
$\dot{N}_i$
$\dot{Q}_0$
Also a port due to its association with mass flow
$\dot{W}$
$\dot{W}$
Work flow
Actual exergy flows — which may classify as fuel or product flows
$\dot{m}$
$\dot{m}$
Mass flow
Actual exergy flows — which may classify as fuel or product flows
$\dot{Q}$
$\dot{Q}$
Heat flow
$\dot{m}$
Mass flow
Actual exergy flows — which classifies as waste flow
Implied system boundary - all exergy destruction is deemed to take place within this boundary
Entering interconnections
Exiting interconnections
Generic exergy flow
Extra-flow transaction
Plant
Cut-point
Storage

Notes: 1. Only one representation of each exergy flow type is shown both entering and exiting. More generally, several may exist, in which case the cut-points need to be indexed.
2. The $\dot{N}_i$ chemical species transactions can only occur in plant which is permeable. The $\dot{Q}_0$ heat transfer can only occur in plant which is non-adiabatic. The $\dot{V}$ work exchange can only occur in plant which is not rigid.
3. The waste flow, by definition, will possess exergy, otherwise the transaction would constitute extra-flow loss.
4. Any exergy service provided by the plant is not shown.

Figure 5.2 *A generalised NSSF exergy process containing storage and based on actual exergy flows. Refer to figure B.2 for the thermodynamic diagram on which this is based. Compare also with figure 8.4 which is based on logical exergy flows.*

adopted in this thesis is due to Kotas (1995, p 73), in which exergetic efficiency is defined, in average and marginal forms, respectively:[10] [11]

$$\eta_{\text{II}} = \left.\frac{\sum_{\text{product}} \Delta\dot{E}}{\sum_{\text{fuel}} \Delta\dot{E}}\right|_{\substack{\text{both relative}\\ \text{to the local}\\ \text{dead-state}}} \qquad \eta'_{\text{II}} = \left.\frac{\partial \sum_{\text{product}} \Delta\dot{E}}{\partial \sum_{\text{fuel}} \Delta\dot{E}}\right|_{\substack{\text{both relative}\\ \text{to the local}\\ \text{dead-state}}} \tag{5.15}$$

where: η_{II} is the average exergetic efficiency
η'_{II} is the marginal exergetic efficiency
$\Delta\dot{E}$ is the change in flow exergy of the flow in question

and: the $\dot{E}$ have the appropriate formulation (see box on p 56)

For terminating flows, originating flows, and logical flows: $\Delta\dot{E}$ simply equals $\dot{E}$.

Equation (5.15) can be presented more descriptively as:[12]

$$\eta_{\text{II}} = \frac{\text{'exergy utilised'}}{\text{'exergy used'}}$$

The definitions given in eqn (5.15) requires two conditions be fulfilled: first, the system can be approximated as SSSF, and second, the commodity flows can be classified by role — which means that the purpose of the system in question must be known prior to analysis. Exergetic efficiency can be applied to components, single plant, or to any contiguous collection of plant — by virtue of the fact that exergy is an extensive quantity.

Exergetic efficiency provides a measure of how closely a finite-time process approaches ideal behaviour. Exergetic efficiency is not necessarily independent of the prevailing dead-state definition in cases where the plant interacts chemically with the environment (*eg*, discharges to the atmosphere).

First law-based SSSF plant efficiency

The first law-based index adopted in this thesis is energetic efficiency and is defined, in average and marginal forms, respectively:

[10] The vertical line notation indicates that condition noted must prevail for the expression to hold.

[11] Other definitions for exergetic efficiency exist. Krakow (1991, p 331) records eqn (5.15) as 'exergy efficiency' as defined in his eqn 30, and the equation below as 'second law efficiency' as defined in eqns 13–15. Krakow makes no attempt to equate these two efficiencies, yet they may well be equal in general.

$$\eta^{*}_{\text{II}} = \frac{\eta_{\text{I}}}{\eta_{\text{I,reversable}}} = \frac{\text{COP}}{\text{COP}_{\text{reversable}}}$$

[12] Given by Wall (1998, p 17).

$$\eta_{\mathrm{I}} = \frac{\sum\limits_{\text{product}} \Delta\dot{L}}{\sum\limits_{\text{fuel}} \Delta\dot{L}}\Bigg|_{\substack{\text{both relative}\\ \text{to the local}\\ \text{dead-state}}} \qquad \eta'_{\mathrm{I}} = \frac{\partial \sum\limits_{\text{product}} \Delta\dot{L}}{\partial \sum\limits_{\text{fuel}} \Delta\dot{L}}\Bigg|_{\substack{\text{both relative}\\ \text{to the local}\\ \text{dead-state}}} \tag{5.16}$$

where: η_{I} is the average ϵnergetic efficiency
η'_{I} is the marginal ϵnergetic efficiency
$\Delta\dot{L}$ is the change in flow ϵnergy of the flow in question

and: the $\dot{L}$ have the appropriate formulation (see box on p 56)

For terminating flows, originating flows, and logical flows: $\Delta\dot{L}$ simply equals $\dot{L}$.

Equation (5.16) can be presented more descriptively as:

$$\eta_{\mathrm{I}} \quad = \quad \frac{\text{'ϵnergy utilised'}}{\text{'ϵnergy used'}}$$

ℑnergetic efficiency can only be applied meaningfully to plant or collections of plant, but not to components.

Depending on the nature of the plant involved, ϵnergetic efficiency, like exergetic efficiency, may be dependent on the dead-state definition, beyond T_0.

NSSS plant efficiency

More generally, NSSF performance indexes, similar to the SSSF indexes identified earlier, can be formulated for NSSF plant. Such expressions are not pursued here but they are used within *deeco* to describe the behaviour of storage plant.

5.5.3 Exergy quality

Flow exergy quality factor

The issue of exergy quality mismatch as a flag for poor network integration and/or poor plant inefficiency was raised earlier (see p 16). In this thesis, the exergy quality factor Z of a given commodity flow is defined as:[13]

$$Z = \frac{\dot{E}}{\dot{L}}\Bigg|_{\substack{\text{both relative}\\ \text{to the local}\\ \text{dead-state}}} \tag{5.17}$$

where: Z is the flow exergy quality factor, subscripted using the scheme adopted for $\dot{E}$ and $\dot{L}$

and: the $\dot{E}$ and $\dot{L}$ have the appropriate formulation (see box on p 56)

[13] Wall (1998, p 55) labels this concept the 'exergy factor' but defines it less generally.

Equation (5.17) can be presented more descriptively as:

$$Z = \frac{'\text{work production potential}'}{'\text{heat production potential}'}$$

Equation (5.17) can be more specifically formulated as follows, using the exergy and ϵnergy formulations given earlier (see eqns (5.1) – (5.12)):

$$Z_{\text{x}} = \frac{(h - h_0) + \frac{(\upsilon^2 - \upsilon_0^2)}{2} + g(z - z_0) - T_0(s - s_0)}{(h - h_0) + \frac{(\upsilon^2 - \upsilon_0^2)}{2} + g(z - z_0)} \tag{5.18}$$

$$Z_{\text{t}} = \frac{(\bar{h} - T_0\bar{s}) - \sum_{i=1}^{n}(\bar{h}_{0,i} - T_0\bar{s}_{0,i})x_i}{\bar{h} - \sum_{i=1}^{n}\bar{h}_{0,i}x_i} \tag{5.19}$$

$$Z_Q = 1 - \frac{T_0}{T} \tag{5.20}$$

$$Z_W = 1 \tag{5.21}$$

Typical values for Z are given in table 5.4. Electricity is a form of work, hence the unity value. The Z values for oxidisable fuels are from Bejan (1997, p 362) and Stepanov (1995). The geothermal example is based on a dead-state of liquid water.

Thermal radiation is not a classical phenomenon and there is some dispute in the literature as to whether the Carnot efficiency ($\eta_{\text{Carnot}} = 1 - T_0/T$) between source and sink can be used to determine Z in this case — several alternative expressions have been proposed with some taking source geometry into account. In the case of terrestrial solar insolation, all give numerical values around 0.93. In addition, clouds, atmospheric turbidity, and sun elevation all tend to reduce the ϵnergy intensity of sunlight, rather than its exergy quality. For more detail, consult: Bannister (1991, p 275–278), Kabelac and Drake (1992).

Fuel flow	**Form**	**Flow exergy quality factor**
electric power	*W*	1.00
sunlight (thermal radiation)	—	~ 0.93
oxidisible fuels	t	0.90 – 0.95
geothermal at 95˚C extraction	x	0.12
space ‘heat’ at 10˚C above ambient — flow equivalent of same	*Q*	0.03

Table 5.4 *Typical flow exergy quality factors for common fuel streams and uses.*

Nonflow exergy quality factor

The nonflow exergy quality factor can also be defined for stock commodities:

$$Z' = \frac{\Xi}{\Lambda} \Bigg| \begin{array}{l} \text{both relative} \\ \text{to the local} \\ \text{dead-state} \end{array} \tag{5.22}$$

where: Z' is the nonflow exergy quality factor, subscripted using the scheme adopted for Ξ and Λ

and: the Ξ and Λ have the appropriate formulation (see box on p 56)

As before, eqn (5.22) can be presented more descriptively as:

$$Z' \quad = \quad \frac{\text{'work production potential'}}{\text{'heat production potential'}}$$

5.5.4 Quality matching

Two-connection plant

Much of the conversion and transport plant in a typical ESSS is two-connection, *ie*, containing one fuel flow input and one product flow output — with these flows being actual or logical as appropriate. For such plant operating close to steady-state and without significant extra-flow losses (*ie*, the process is constant volume, impervious, and adiabatic, $\dot{Q}_0 = 0$) the exergetic efficiency can be found as follows: substitute eqn (5.17) into eqn (5.15) and note that $\dot{L}_{\text{fuel}} = \dot{L}_{\text{product}}$ to yield:

$$\eta_{\text{II}} = \frac{Z_{\text{product}}}{Z_{\text{fuel}}} \qquad \text{(adiabatic two-connection plant)} \tag{5.23}$$

In this case, the εnergetic efficiency is always unity, so it is the exergetic efficiency that gives the more informative index.

Design implications

The design implication which arises from eqn (5.23) is that two-connection plant with significant exergy quality mismatch, that is $Z_{\text{fuel}} \gg Z_{\text{product}}$, can, in all likelihood, be reconfigured to advantage by either:

- proposing a different two-connection technology requiring lower quality fuel and casting around for a suitable fuel source within the network,
- placing a three-connection work-producing process immediately upstream and casting around for a suitable load to supply within the network, or
- placing a two-connection service-producing process immediately upstream and casting around for a suitable exergy-service demand within the network.

In addition, any gain in plant efficiency should, in most instances, reflect positively on sustainability, irrespective of where the plant is located within the network. In all cases, the proposed changes would need to be confirmed by optimisation modelling, particularly if the embodied sustainability costs are significant.

5.5.5 Approximating real processes as multi-connection plant

Generalised multi-connection plant

Figure 5.3 shows two forms of plant, one SSSF and the other intermittently NSSF. As discussed in chapter 7, the connections, in effect, represent the interfaces between abutting process.

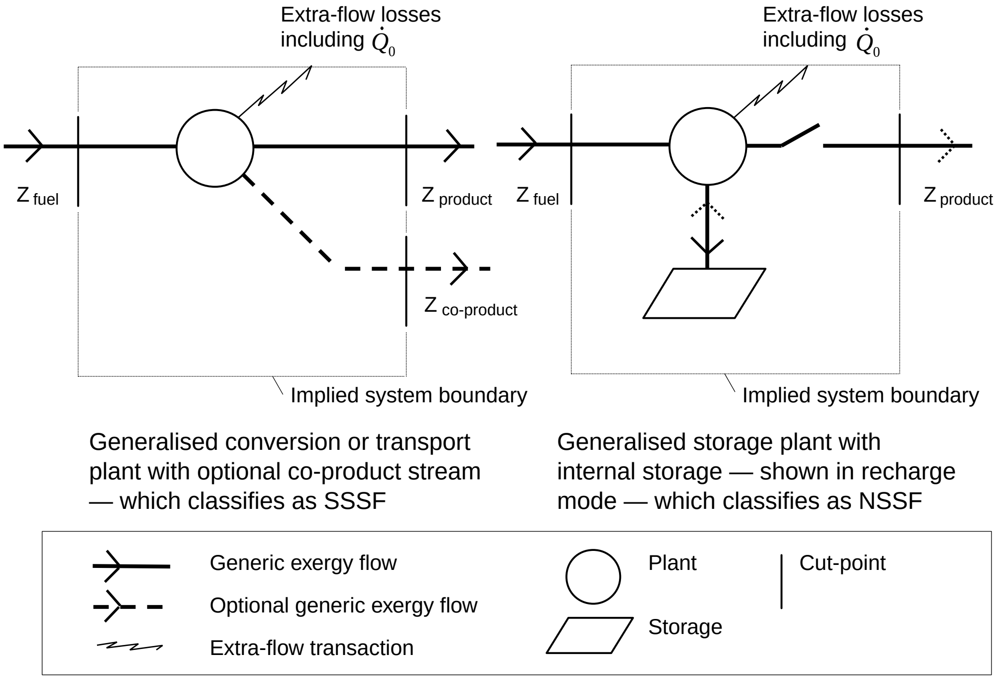
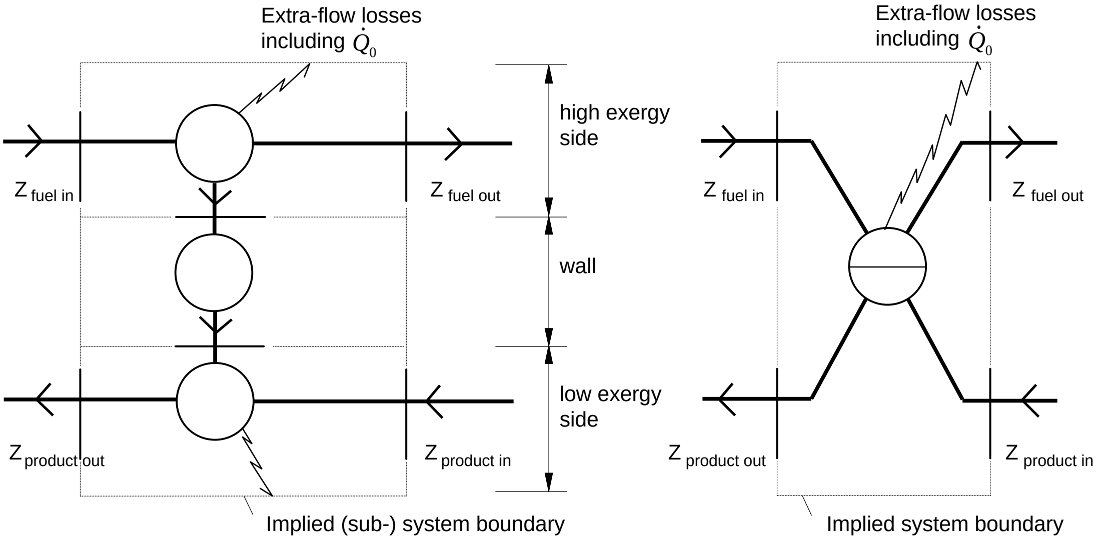

Extra-flow losses including $\dot{Q}_0$
Extra-flow losses including $\dot{Q}_0$
Z fuel
Z product
Z co-product
Z fuel
Z product
Implied system boundary
Implied system boundary
Generalised conversion or transport plant with optional co-product stream — which classifies as SSSF
Generalised storage plant with internal storage — shown in recharge mode — which classifies as NSSF
Generic exergy flow
Optional generic exergy flow
Extra-flow transaction
Plant
Storage
Cut-point

Figure 5.3 *Generalised steady-flow process representations for some given time-interval based on generalised multi-connection plant.*

The three-connection conjecture

It is conjectured that any given contiguous set of NSSF thermodynamic processes can be reduced to a local network of two and three-connection plant of the types shown in figure 5.3. However for certain processes, the discrete plant would need to become vanishingly small in order to create a piecewise continuum approximation. In such cases, a better strategy is to adopt a four-connection or higher connection-count plant representation, even though this will result in some internal detail being lost.

This issue can be readily illustrated by representing a standard counter-flow heat-exchanger as three and four-connection plant as shown in figure 5.4. As a consequence of moving to a four-connection model, some internal structure is lost — and this reduction in information leads to a reduced ability to pinpoint the location and form of the exergy destruction mechanisms at play.



Three-connection plant representation assuming invariant stream temperatures. If the stream temperatures vary with flow direction, then a stream-wise discrete approximation would be required

Four-connection plant representation which is entirely general, but which captures less of the internal structure

Notes: 1. Actual rather than logical exergy flows are used.
2. The horizontal line on the plant symbol indicates the absence of stream mixing.

Figure 5.4 *A standard counter-flow heat-exchanger approximated using networked multi-connection plant. The four-connection description may prove more tractable but information is lost in the reduction.*

However, providing a proof for the ′three-connection conjecture′ is beyond the scope of this thesis, and furthermore the network optimisation model under consideration does not require this assertion to be true for its validity.

5.6 Abstractions required for the flow network model

Three further abstractions required for the flow network model are described at this point: logical plant, logical exergy flow, and abstract sourcing plant and demand plant. But first, the notion of cyclic structures is introduced, because the thermal sub-network algorithm (see p 136) used by *deeco* requires a cycle-free graph representation.

5.6.1 Cyclic structures

A cyclic structure occurs when an actual exergy flow path involving two or more plant (or their components, if the analysis descends to this resolution) reconnects with itself. Cyclic structures are, in most cases, not tenable for the model under disussion because the attributes of the exergy flow in the various links cannot be resolved. Cyclic structures are typically encountered in a real ESSS in one of five contexts:

- work buses,
- component connectivity within cyclic plant,
- waste-heat reuse,
- convective heat exchange between neighbouring plant,
- re-circulating work exchange between neighbouring plant.

By way of example, figure 5.4 depicts a plant with fuel and product flows which both enter and exit — in this case, either the second or fourth situation could apply.

Long range cyclic structures are seldom encountered in an ESSS in practice — such structures are self-evidently wasteful, and in addition, may degrade network stability. The following sub-sections expand on the five points just raised.

Work buses

Work buses or work pools occur when work in one of its forms (*eg*, electrical, mechanical) is re-injected into a common bus or layshaft, as the case may be. Work buses are legal within the model under consideration, but the optimisation procedure will seek to limit the ′recycling′ of work in cases where additional selected flow costs arise.[14]

Logical plant

The internal connectivity of cyclic plant (see p 169) is not tractable under the network optimisation model under discussion. Such plant instead needs to be captured and replaced by a single logical plant entity. This strategy hides the cyclic structures from view, but also means that the internal arrangement of the cyclic plant is no longer accessible for optimisation (should this have been possible). The logical plant description is also used to represent plant which produces useable waste-heat, as described elsewhere.

[14] *deeco* explicitly supports a regional electricity pool.

Logical exergy flow

In practice, neighbouring plant often exchanges heat via forced convective heat transfer, or work via mechanical means (*eg*, hydraulic systems). In either case, *flo* and *return* pathways (in the most general sense) exist, which necessarily creates a cyclic structure between the two plant in question. The solution is to represent this arrangement by a single interconnecting flow, termed a 'logical exergy flow'. The concept of logical exergy flow also includes 'singly-connected' actual exergy flow as a natural subset.

An example of the mapping of two actual exergy flows to one logical exergy flow is shown on p 126. Within *deeco*, the logical flow abstraction is principally used to capture convective heat transfer involving *flo* and *return* transfer media pipework — with the attributes of the logical heat flow resolved by the thermal sub-network (TSN) algorithm. The logical exergy flow method can be applied where the cyclic structure directly extends beyond the two neighbouring plant, but only in certain circumstances (see p 146).[15]

5.6.2 Demand plant and sourcing plant

The flow network model under consideration is based on information concerning demand for service and concerning sourcing offers. This information is translated into exergy flows by the two types of abstract plant indicated in figure 5.5. In essence, the demand plant 'drive' the network, and the sourcing plant 'provide' the network with the reserve of exergy. Figure 1.1 (see p 9) shows sourcing plant and demand plant located on the network — as depicted on the right- and left-sides, respectively.

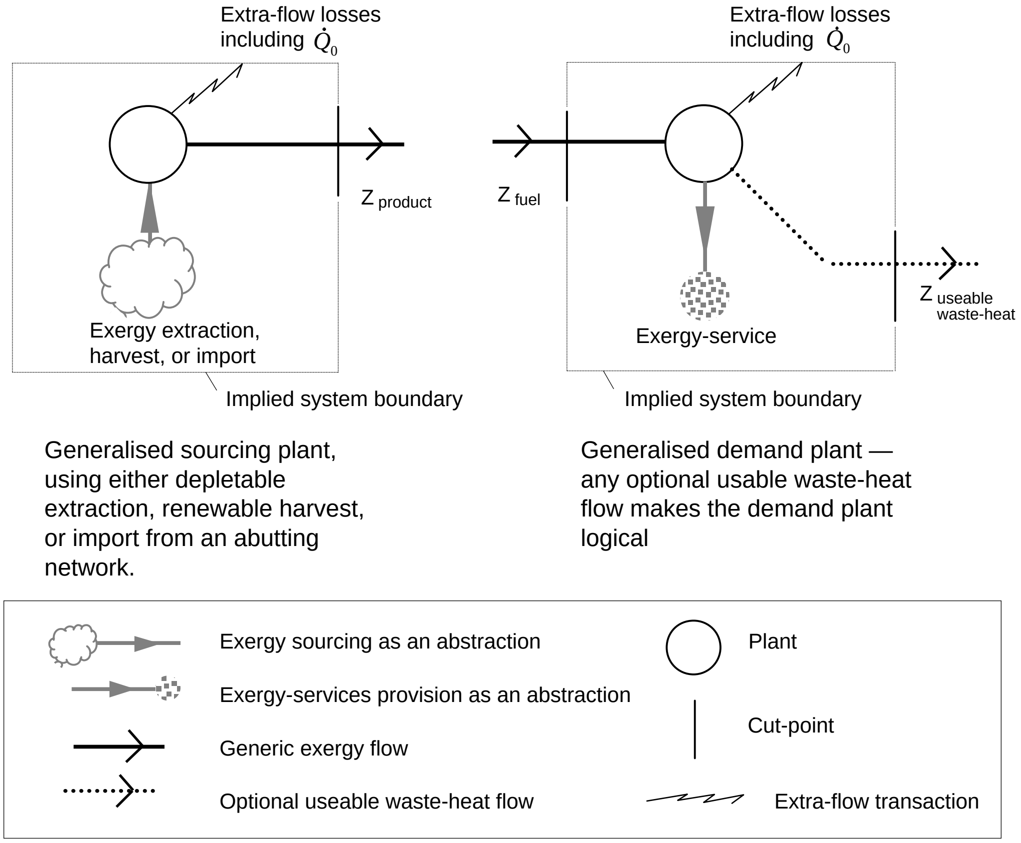
5.7 $nergy accounting

The final section in this chapter reviews $nergy accounting in theory and practice — a topic which falls under the wider discipline of energy economics.

$nergy

In this thesis, $nergy is taken to be a multi-dimensional concept (*ie*, a vector), bundling truncated exergy content [J] (explained shortly), price [$], and additional attributes as appropriate.

[15] The logical exergy flow method could perhaps be extended to include chemically active processes — in cases whereby a receiving plant returns its reaction or diffusion products back to an originating plant for reinstatement. The extent to which *deeco*-like techniques could have a role in chemical as opposed to thermal plant analysis remains an open question.


Extra-flow losses including $\dot{Q}_0$
Extra-flow losses including $\dot{Q}_0$
$Z_{product}$
Z_{fuel}
$Z_{useable\ waste\text{-}heat}$
Exergy extraction, harvest, or import
Exergy-service
Implied system boundary
Implied system boundary
Generalised sourcing plant, using either depletable extraction, renewable harvest, or import from an abutting network.
Generalised demand plant — any optional usable waste-heat flow makes the demand plant logical
Exergy sourcing as an abstraction
Exergy-services provision as an abstraction
Generic exergy flow
Optional useable waste-heat flow
Plant
Cut-point
Extra-flow transaction

Notes: 1. Under *deeco*, a demand process providing usable waste-heat flow needs to be modelled using two separate processes. This complication is not indicated in the diagram, but means that the demand plant shown would need to be logical, containing an actual demand plant and a sourcing plant for the waste-heat. This means useable waste-heat flow may be re-injected upstream, unlike other forms of non-work co-product.

Figure 5.5 *Abstract processes used to source exergy flow or demand exergy flow within the flow network model. Refer also to figure 1.1.*

$nergy is essentially an economic construct. $nergy may be demanded as either a factor of production or as a public or private factor of consumption — these latter forms arise to satisfy some human need or desire, albeit indirectly. Differences which arise from these various purposes are considered immaterial, and $nergy is assumed to be priced consistently irrespective of role.

Any physical commodity flow can be viewed as a generic factor of production or consumption, yet the flow can only be classified as $nergy *if and only if* the flow is *valued* for its exergy content and production convenience above all other alternative uses. Furthermore, the physical characteristics of the flow are of interest only to the degree to which the commodity meets operational requirements, or, framed in economic terms, the degree to which these impact on its substitutability — with substitutability being indicated by a high cross-elasticity of demand with alternative forms of supply. Hence, the analyst is *required* to explicitly assess the economic worth of the commodity flow *before* classifying it as $nergy or otherwise — and before accounting for a given commodity flow. This approach is in strong contrast to that used by engineering systems analysis, whereby every exergy contribution must be counted, irrespective of role.

In the first instance, the price is assumed to reflect the exergy content and the production convenience of the $nergy flow at the point of economic demand (van Gool 1987). But this view is oversimplistic, given that most $nergy is supplied using reticulation grids, and that grid membership generally provides significant value in its own right — for example, increased security of supply and reduced variability in product quality. The question of determining the value that grid membership confers is beyond the scope of this thesis.[16]

Hence, $nergy is defined using at least two components — the first is thermodynamic, the second economic, and any remainder will quantify aspects of production convenience (*eg*, sulphur content). The thermodynamic component of $nergy is evaluated using truncated forms of either flow exergy or flow ϵnergy by omitting those terms with no economic value. Therefore, $nergy requires that both the technological and market contexts be considered in addition to the thermodynamic environment context. Observations indicate that the exergy formulation more closely aligns with economic value, hence:

$$\dot{\Sigma}^E \leq \dot{E} \tag{5.24}$$

where: $\dot{\Sigma}^E$ 'energy' rate component of Σ accounted using flow exergy
$\dot{E}$ flow exergy rate using the appropriate formulation
(see box on p 56)

[16] Network economics attempts to include the implications of grid structure within microeconomic supply and demand relationships. Nonetheless the issue is a research field and the writer could locate very little on this topic in the literature.

Therefore, the appropriate formulation for flow exergy should be applied at the cut-point, but in this case only those terms exhibiting economic value should be included, this being indicated by the inequality in eqn (5.24). For historical reasons national $nergy accounting protocols have been based on flow ϵnergy accounting, hence:

$$\dot{\Sigma}^L \leq \dot{L} \tag{5.25}$$

where: $\dot{\Sigma}^L$ 'energy' rate component of Σ accounted using flow ϵnergy
$\dot{L}$ flow ϵnergy rate using the appropriate formulation (see box on p 56)

Similarly, the appropriate formulation for flow ϵnergy is required. $nergy accounting and the deficiencies which arise from current practice is discussed in section 4.2.1.

Summarising the previous discussion in mathematical terms:

$$\Sigma = (\Sigma^{E|L}, \Sigma^{\$}, \Sigma^{\mathrm{A}}) \tag{5.26}$$

where: $\Sigma^{E|L}$ indicates either Σ^E or Σ^L as appropriate
$\Sigma^{\$}$ is the factor price
Σ^{A} indicates one or more factor attributes (*eg*, sulphur content)

Problems associated with non-market valuations, externalities, and incomplete and uncertain information as well as with network-effect market failure have been set aside in order to progress this discussion just given. These shortcomings do not present difficulties for the principal themes in this thesis, as these do not rely on the concept of $nergy.

Emergy

The concept of emergy is not used but does warrant mention. Emergy attempts to account for the solar insolation investment in a given fuel flow. Emergy is based on the premise that bulk of the exergy used by industrial society derives from sunlight and that fossil fuels can be interpreted as stored solar insolation. Emergy advocates have developed an emergy theory of value, analogous to the labour theory of value articulated under classical economics. (Odum 1994).

5.8 Closure

Several key abstractions, necessary to establish the flow network optimisation model, have been introduced thus far: the bulk commodity approximations, the multi-connection plant abstraction, logical exergy flow and logical plant, and source plant and demand plant abstractions. Chapter 6 describes the generic flow network optimisation problem and chapter 7 then combines these themes.

The flow and nonflow ϵnergy functions are necessary when dealing with resource assessment, and for first law process performance evaluation when mass streams interact directly with the environment. An analogous situation applies to flow and nonflow exergy functions. Placing both suites of functions on an identical footing, as has been attempted here, allows for methodological compactness and consistency of application. In addition, most of the methodological criticisms directed at exergy analysis apply equally to ϵnergy accounting — avoiding the second law does not remove the methodological difficulties posed by environmental dependence and dead-state definition in a non-equilibrium world. □

Chapter 6

Physical networks and flow network optimisation techniques

This chapter backgrounds the flow network optimisation theory relevant to deeco.

6.1 Introduction

This chapter focuses on network theory. First, networks are defined and classified, and some network analytical techniques are outlined. Second, a selection of real-world network applications are reviewed, so that, where possible, insights from this work could be included. Third, the flow network optimisation problem is formally structured as the ′minimum cost flow problem′ or MCFP. Fourth, methods for solving the MCFP are reviewed. Fifth, extensions to the basic MCFP are outlined, as these may help overcome methodological problems or prove useful in the event of further development for *deeco* itself. Finally, dynamic programming and multiple objective optimisation are summarised, again for future reference.

6.2 Generic concepts

6.2.1 Networks

A network is a structure comprising discrete vertices and directed edges.[1] As such, a network is an abstraction, but one which can usefully model a wide range of discrete physical and nonphysical phenomena. Further, it may be convenient to assign sets of properties to edges and vertices. Figure 6.1 shows a directed graph with weighted edges.

Networks can be viewed in two ways. A topological interpretation is concerned

[1] Vertices were termed ′nodes′ and edges were termed ′links′ in earlier sections of this thesis.

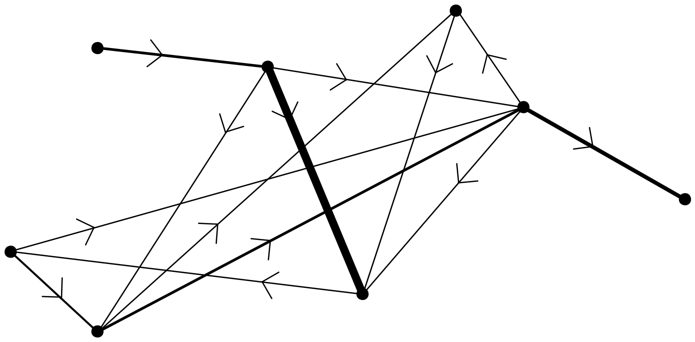


Figure 6.1 *Directed graph with weighted edges as indicated by line thickness. The graph shown is also nonplanar.*

only with the interrelation between vertices and edges. A geometric interpretation, on the other hand, requires the edge and vertex properties to be included. Hence, geometric congruence is more demanding than that topological equality.

6.2.2 Flow networks

If the edges represent commodity flows of some description, the network is termed a ′flow network′. The vertices then represent flow junctions, which are also barred from accumulating flow. If flow costs and flow bounds can be associated with each edge, then the flow regime may be amenable to constrained optimisation. Flow networks typically contain one or more source and sink, in which case commodity demand can be represented using sink flows.

Commodity flow types include the flow of funds, objects, data (*eg*, as bit streams, cells, or packets), charge, and mass, or any variable expressible as the time-rate of change of some extensive quantity.[2] This definition of commodity does not necessarily align with that used in economics.

6.2.3 Physical networks

Flow networks may also contain flow-modifying components located on their edges.[3] Such networks are termed ′physical networks′ (Blackwell 1968, Jensen and Barnes 1980). Physical networks always carry exergy in some form, but the flow itself may be valued for its ability to convey data, its material properties, or for its exergy content. Alternatively, some network characteristic may be valued, such as high attenuation or retarded transport — as provided by acoustic, electrical, or thermal

[2] Bruckner (1997, translation) limits commodities to those entities for which a conservation law exists — therefore excluding exergy and entropy.

[3] The term ′component′ is used in this chapter in the same sense as that given in appendix A.

insulation. Neither the commodity flow nor the network attributes can be considered in isolation, rather the respective importance depends on the purpose of the network.

In the simplest case, the network carries just one commodity type, and its components are two-port (*ie*, two-terminal).[4] Active components can be interpreted as forcing the flow — they will, in turn, need to be gatewayed to another network for their power. Passive components either destroy exergy in the flow, or transiently accumulate and release exergy into the flow, or do both. In addition, it is never fully possible to isolate a network, so that unwanted coupling between the network and its surroundings may result in significant losses (*eg*, long distance AC transmission can exhibit significant shunt losses).

Component behaviour can be specified using component ′characteristic equations′, which relate conditions on the component terminals.[5] Such equations can be established using empirical measurements, thus treating the component as a black-box, or alternatively they can be derived from analytical arguments. If the components are linear, the characteristic equations may be expressed as transfer functions in the Laplace (s) or z-transform (z) domains.

Networks may contain more that one commodity type, in which case four-port (*ie*, four terminal) components are required to interface any two single-commodity constituent networks. Thus, some feasible transformation process is required to in order to gateway the two constituent networks. However, this situation is not the same as a ′multi-commodity network′ whereby two or more commodity types can carried on the same edge.[6]

Networks may be open or closed in terms of mass flow. Open networks transfer the network commodity from sources to sinks. Closed networks, on the other hand, transfer either work or heat from one network interface to another without requiring external mass flow (*eg*, electrical circuits, hydraulic systems, layshafts, and heat-exchanger networks). Certain closed networks are occasionally described as ′energy′ vectors.

Kirchhoff's laws

One of the principle tools for analysing isolated physical networks are Kirchhoff's laws.[7] Kirchhoff's ′voltage′ law holds that the sum of forces (potential differences) in any cycle in the network is zero. Kirchhoff's ′current′ law holds that the sum of flows (currents) at any vertex in the network is zero. Kirchhoff's laws apply at the

[4] The term ′one-port′ is also used to indicate the same arrangement, so care is needed.

[5] *deeco* uses the term ′input-output equation sets′ instead of characteristic equation.

[6] Foulds (1979) discusses multi-commodity networks.

[7] Although accorded the status of laws, Kirchhoff's laws may also be described as postulates.

topological rather geometric level.[8] Kirchhoff's laws can be given in mathematical terms as follows.

First, the concept of potential is assumed to precede that of force, so that vertex i is assigned potential e_i. One vertex is usually given zero potential and becomes known as the ground. Hence, the force X_k in edge k is:

$$X_k = e_{\text{tail of } k} \quad - \quad e_{\text{head of } k} \tag{6.1}$$

Kirchhoff's ′voltage′ and ′current′ laws are then given as, respectively:

$$\sum_{\text{cycle}} X_k = 0 \qquad\qquad \sum_{\text{vertex}} J_k = 0 \tag{6.2}$$

where: X_k represents the force (potential difference) in edge k
J_k represents the flow (current) in edge k

Note that X_k is intensive and J_k is extensive. Using the same notation, the component characteristic equation G_k for edge k is of the form:

$$J_k(t) = G_k(X_k(t)) \tag{6.3}$$

Physical networks, taken generally, thus involve forces and flows. Which of these two quantities is *a priori* primitive and which is the resultant is not a question that needs resolution before analysis can proceed — due to the fact that Kirchhoff's two laws lie in orthogonal subspaces (Peusner 1986). Kirchhoff's laws can be interpreted as the network equivalents of a system force balance and a system mass or charge balance as appropriate.

Analysis of planar physical networks

Planar networks are networks where no two edges need cross when the network is drawn on a sphere.[9] Polynomial time algorithms exist to test for planarity by examining the geometric dual of the network. Furthermore, it can be shown that any planar network can be drawn on a flat surface using only straight lines which do not cross (Foulds 1994).

Analysis of planar physical networks proceeds using Kirchhoff's two laws and the component characteristic equations. Normally, at least one component is active. A spanning tree is used to identify a set of fundamental circuits for Kirchhoff's

[8] Peusner (1986) notes that Kirchhoff's laws have been generalised as Tellegen's theorem. Dolan and Aldous (1995) establish Tellegen's theorum for restricted cases.

[9] More formally, the network can be embedded on a surface of genus zero. Nonplanar networks require higher genus embedding surfaces. The complete graph K_5 and the complete bipartite graph $K_{3,3}$ are both nonplanar.

'voltage' law, and a set of fundamental cut-sets for Kirchhoff's 'current' law.[10] [11] These equations together with the component equations can be solved analytically, symbolically (*ie*, using computer algebra), or numerically to give a dynamic description of the network. (Blackwell 1968). The writer understands that no such general method exists for nonplanar physical networks.

Smith (1982) gives a good introduction to generic flow networks and their optimisation, and Blackwell (1968) does likewise for generic physical networks and their analysis.

6.2.4 Related issues

Resource-distributing networks in nature

The physical networks just described are usually interpreted as technological. However, resource-distributing networks are ubiquitous in nature, and these include circulatory systems in animals and vascular systems in plants. These tree-like networks exhibit the fractal property of structural self-similarity over a considerable range of scales. West, Brown, and Enquist (1997) suggest that resource-distributing networks have two remarkable characteristics. First, numerical simulation suggests that the fractal structure observed represents the optimal arrangement for distributing resources in terms of efficacy versus loss. Second, that limitations imposed by network performance rather than aggregate energy balance controls the size of organisms as they scale from small to large under a 3/4 power law.

Most exergy-services supply networks clearly exhibit a similar fractal-like morphology. The question arises whether the conclusions of West *et al* (1997) are relevant to this project. Unfortunately, this question must be left open — due to time constraints rather than lack of interest.

Entropy optimisation techniques

Fang, Rajasekera, and Tsao (1997) reframe many of the network problems given in this chapter — including input/output modelling — in terms of entropy optimisation. It would be interesting to interprete the *deeco* model in this context.

Flow intensity

One clear drawback of network models is that flow intensities are not explicitly captured.[12] This poses an added problem in the context of sustainability modelling,

[10] A spanning tree is any tree which covers (*ie*, includes) all vertices on the underlying network. Polynomial time algorithms exist to locate spanning trees.

[11] A fundamental circuit arises by adding a chord to the spanning tree. A fundamental cut-set arises by partitioning the network such that the spanning tree is cut only once.

[12] Flow intensity is flow-rate/unit area, and hence is synonymous with current density.

as non-global environmental effects are often flow-intensity-related. However, generally this loss of precision is a small price to pay for avoiding the complexity of a continuum model. In addition, it may be possible to use network cost functions normalised against cross-sectional area, so that flow cost intensities rather that flow costs are optimised.

6.3 Real-world networks

With very few exceptions, real-world networks deliver services of some description. Service-delivery by network offers many advantages, including, potentially, reduced capital and environmental costs, cheaper primary inputs, and improved quality and reliability. It should be noted that whilst the services themselves are logical, an underlying physical infrastructure and some physical commodity inputs are inescapable.

Networks may fulfil some stand-alone purpose within a single organisation, or they may offer services of value to a number of participants. Participant-based networks, irrespective of type, are defined by their physical layout and by the network standards that their users must comply with in order to connect and draw benefit.[13] Such standards are generally aimed at protecting the collective interest. The benefits of membership typically increase with scale and density while relative costs decline — networks with these characteristics are said to possess some natural monopoly status. Participant-based networks are usually dynamic, both operationally and structurally. In addition there is a clear trend, in New Zealand at least, toward system regulation based on price rather than some predetermined essential service status.

When analysing networks, it is important to capture the entire network and not simply focus on the more visible components such as wires and pipes. Defining the network scope and drawing its bounds for a given problem is generally not trivial and may require judgement on the part of the modeller. Furthermore, the concept of 'network' is wide ranging, so working definitions tend to be specific to a particular class of problem and closely shaped by the mathematics used to describe that class of problem.

6.3.1 Commonly encountered network models

It is useful to review a range of network problems — all possess relevance to *deeco* and several classify as physical (*ie*, they contain flow-modifying components):

[13] Examples of network standards include acceptable use policies (AUP) for computer networks, and system connect agreements for electricity distribution grids.

- electrical circuits (engineering),
- control/process systems (engineering),
- piping layouts (engineering),
- thermal networks (physics),
- heat-exchanger networks (engineering),
- input/output modelling (economics),
- transport and communication networks (operational research),
- flow networks (operational research).

Electrical circuits

The analytical modelling of electrical circuits requires the network be planar so that a system of equations based on Kirchhoff's two laws (see eqn (6.2)) can be identified. The circuit is represented as an arbitrarily directed graph, and each edge contains three entities: a component, a flow of charge, and a potential difference between its ends. As discussed earlier, an efficient method for identifying a set of independent network equations is to fix a spanning tree and use this to identify sets of fundamental cycles and cut-sets. Furthermore, the component characteristic equations need to be linear (or linearised) if the system of equations is to be solved algebraically, otherwise a numerical solution may prove tractable.

Additional simplifications arise if transient responses can be ignored in the case of circuits forced by either DC or by constant frequency AC (*eg*, using phasor methods). Analysis of nonplanar circuits requires the use of analog or numerical simulation techniques.

Blackwell (1968) reviews these and analogous problems. Foulds (1994, p 269–279), succinctly describes electrical circuit analysis in graph-theoretic terms. Jensen and Barnes (1980) reframe DC circuit analysis as a convex cost flow network optimisation problem.

AC circuits in which the wavelength is comparable to the path length requires that the lumped parameter network view be abandoned in favour of a distributed description. Such analysis is covered under transmission line theory (Jensen and Watkins 1974).

Process/control systems

Control systems together with their controlled processes can readily be viewed in network terms. Such systems are typically modelled using block diagrams, with the interconnecting links carrying various commodities including process state data. Analytical modelling is greatly simplified if the system blocks are linear (or linearised), in which case Laplace (s) or z-transform (z) domain transfer functions can be used to determine inherent system stability and to solve for system dynamics. Furthermore, control system performance can be evaluated against some stated criteria, such as RMS deviation or power demand, and subject to optimisation. Seborg *et al* (1989) review process control theory, and Casti (2000, ch 3) describes control system optimisation.

Piping networks

Piping networks carrying incompressible fluid can be modelled by analogy with electrical circuits. Static pressure then equates with voltage, and flow-rate with current. In more complicated cases, empirically-based dimensional analysis can be used, or failing that, continuum mechanics-based computational fluid dynamic (CFD) simulations can be applied.

Thermal networks

Thermal networks describe heat flow along radiation and conduction paths, and can also be modelled by analogy with electrical RC circuits. Temperature (T) equates to voltage and heat flux ($\dot{Q}$) to current. For conduction, heat transfer is assumed linear with temperature gradient (Fourier's law), while for thermal radiation the relationship is highly nonlinear (Stefan-Boltzmann equation).[14] Peusner (1986) generalises the concept of thermal networks to model discretized thermodynamic systems — his treatment is beyond the scope of this thesis.

Heat-exchanger networks are extremely difficult to analyse due to coupling between irreversible loss mechanisms. As a result, viscous loss (pressure drop) is often ignored and the analysis proceeds by considering only finite-time (irreversible) heat transfer. Pinch analysis (see p 48) seeks opportunities to redistribute heat transfer temperature drop more evenly across both normal (*ie*, heat-exchangers) and cyclic plant (*ie*, heat-pumps and heat-engines), with a consequent decline in exergy destruction. Pinch analysis also contains heuristics to guide and rationalise network redesign efforts. Cost-optimal design (see p 49) seeks to identify maldistributed financially-normalised exergy loss by component, but the method has yet to incorporate network optimisation techniques.

Economic input/output modelling

The Leontief open input/output economic model can be represented as a directed network connecting a number of industrial sectors with one point of final multiple

[14] It is interesting to speculate that a resistance network should also adopt the flow configuration that results in the lowest rate of entropy generation, consistent with transport laws applicable to the particular problem.

demand. The edges carry flows of funds, anti-parallel to the flow of goods. The aim is to predict the commodity flows required to satisfy a pre-determined demand vector, given *a priori* prices for all intermediate and final goods, and the ′value-added′ for each sector.[15] Technology is specified using technology coefficients, which remain invariant. The model is steady-state and does not allow for stockpiling — in other words, the model is SSSF. Production is assumed free from economies or diseconomies of scale. A given problem can be proved meaningful and solvable by locating a spanning out-tree rooted in the final demand.[16] The problem proper can then be solved using either linear algebra or graph algorithms.

This type of model is often subject to comparative static analysis whereby a single exogenous factor is varied and the model rerun. Extensions to this model include Ponsard's model of inter-regional equilibrium. Avondo-Bodino (1979, p 247–251) and Foulds (1994, p 199–202) both review the inter-regional problem, which has clear similarities with the model that underpins *deeco*.

Flow networks

Transport and communication networks can readily be viewed as destination-specific flow networks. Edges represent transport routes or communication lines, and edge weightings represent either distance or cost.

Commodity networks can be viewed as non-destination-specific flow networks. Networks of this type often these carry a single commodity, for instance: water, gas, or electricity.

Flow networks are, in general, amenable to a surprisingly wide range of classification and optimisation procedures. Flow networks can also be used to model assignment and scheduling problems. In the latter case, time is used as the flow cost. These applications are of limited interest here, except to note that Gantt scheduling charts provide a useful way to visually depict networks.

6.3.2 Some observations

Although universal statements about networks are difficult to make, some observations relevant to this thesis which arise from the preceding material are worth recording.

From an engineering perspective, physical networks can be defined as systems for which a "lumped system" approximation is valid (Blackwell 1968, p 2). In other

[15] ′Value-added′ is the price difference between goods shipped and goods required for production. Equating value with price is a fraught affair, in which case ′price-added′ would be more accurate.

[16] An ′out-tree′ is a directed tree with all edges pointing away from the root vertex. An out-tree is also known as an arborescence (see p 150).

words, a useful level of discretisation can be introduced in order to simplify what is ultimately a continuum problem in the real-world. Peusner (1986) develops this theme for discrete thermodynamic systems, arguing the network method yields effective insights.

Systems analysis can be either static, dynamical[17] in the sense of iterated static, or dynamic. Iterated static systems require a discretized time interval that is long when compared with the system relaxation time. Dynamic analysis is not necessarily more insightful than the other forms, and may be less tractable to optimisation.[18]

Most participant-based networks are actively managed by some external agency (*ie*, the network operator) in order to meet certain objectives and requirements.

Data flow (*eg*, analog signals, sampled-data-streams, bit-streams, and hybrids) is generally taken to be exempt from the requirement for flow conservation. In other words, the data-stream can be repeatedly divided without problem.

Networks are invariably represented using mathematical graphs. Topics in graph theory and discrete optimisation constitute the remainder of this chapter.

6.4 Graphs

6.4.1 Graph theory

The study of mathematical graphs is the domain of graph theory.[19] In formal terms, a graph $G = (V, E)$ consists of a finite nonempty set $V(G)$ of vertices, and a set $E(G)$ of edges comprising pairs of different vertices. A directed graph has a direction associated with every edge.

Graphs can be depicted in $\mathbb{R}^2$ space using points to represent vertices, and interconnecting lines to represent edges. If necessary, arrowheads can indicate edge direction, and line weight can indicate edge weighting, as shown in figure 6.1 (see p 77).

Graphs often provide a convenient structure for representing relationships between discrete entities. Graphs have been used to model a wide variety of real-world situations, some interpret directly whilst others are more abstract. Within graph theory, formal proofs and working algorithms relate closely, thus making the discipline particularly suitable to the numerical processing of problems. Many of the algorithms seek some optimal configuration or matching.

In the case of flow networks, vertices represent flow junctions, edges represent the

[17] See footnote 10, p 5

[18] Dynamic analysis should not be confused with dynamic programming (see p 98).

[19] The graph theory terminology used here is generally consistent with Gibbons (1985), most of which he defines on p 1–8. In addition, see the definitions given on p 117 here.

lines along which the chosen commodity flows, and the edge weightings often represent flow costs.

In the case of physical networks, vertices represent flow junctions, edges represent components, and four-port components can be used to interface networks carrying different commodities.[20]

Ore (1990) introduces classical (non-numerical) graph theory. Gibbons (1985) gives a computer science-based treatment. Dolan and Aldous (1995) and Foulds (1994) concentrate on applications. Winter (1989) characterises network problems and solution methods. Smith (1982) and Jensen and Barnes (1980) concentrate on flow network optimisation. Derigs (1988) provides an excellent formal account of flow network problems as well as matching problems. Nemhauser and Wolsey (1988) present discrete optimisation theory and include material on graph routines. Golden and Magnanti (1977) summarise the deterministic network optimisation literature. Finally, the review by Casti (1996, ch 5) is particularly suited for those wanting a lay introduction to the mathematics underpinning this thesis.

6.4.2 Graphs as data-structures

In computer science terms, a graph is a data-structure comprising the vertex and edge sets previously mentioned, together with a suite of useful procedures, such as vertex and edge traversal. Graphs are generally implemented as dynamic structures and are often used as containers. Trees and lists can be viewed as special cases of graphs. (Aho, Hopcroft, and Ullman 1983).

Open-source and proprietary compile-time libraries supporting graph classes are available for a number of high level programming languages including C++ and Java. *deeco* makes use of one such proprietary graph class library, from the USL Standard Components suite.

Whilst on the topic of data-structures, entity–relationship–attribute (ERA) modelling, which underpins most database programming, potentially could be applied to describe flow networks. Moreover, SQL2 contains limited support for graph-theoretic routines.[21] *deeco* uses a quite different approach, but a review of ERA modelling theory may offer useful insights. Using a relational database management system (RDBMS) for preparing data input for *deeco* is a different, but not entirely disjoint matter, and is covered elsewhere (see p 156).

[20] *deeco* uses the terms 'balance point' for flow junction, and 'process' for component.

[21] SQL is 'structured query language'. SQL2 refers to the language standard ISO/IEC 9075:1992 – International Standard Database Language SQL (1992)

6.4.3 Specification of directed graphs

As noted, flow networks can be naturally represented using directed graphs. A directed graph can be specified using (Foulds 1994, p 146–149):

- an adjacency matrix,
- an adjacency list (often exhibiting a graph traversal-based structure),
- an vertex-edge incidence matrix (as defined by eqn (6.4)),
- an edge listing defined in terms of vertices.

The choice depends on the size and sparsity of the graph under consideration and on the algorithm to be applied. All options allow edge weightings to be encoded if required.

6.4.4 Flow networks

In the following analysis, flow networks are taken to require graphs which are:

- directed,
- connected, *ie*, without regard to edge direction,
- not necessarily planar,
- not necessarily simple, *ie*, with multiple edges but free from self-loops,
- deterministic, *ie*, edge attributes are not defined using probability functions.

Some authors also require graphs representing flow networks to:

- be free from directed circuits,
- contain exactly one source and sink.

These two attributes are not necessary for the discussion presented in the next section.

The remainder of this chapter reviews flow network optimisation theory. It is this theory that underpins *deeco*.

6.5 Flow network optimisation

Flow network optimisation falls under the branch of mathematics called discrete optimisation. Within this field, several seemingly distinct mathematical vantage points can be used to formulate a given network flow optimisation problem. But before reviewing the mathematical tools available, it is useful to examine the flow optimisation problem in more detail.

6.5.1 Minimum cost flow optimisation problem (MCFP)

The most general form of the single-commodity flow optimisation problem is the ′minimum cost flow problem′ or MCFP. The commodity must be fungible and divisible.[22] This form of problem is the one tackled by *deeco*.

MCFP can be equivalently defined using networks comprising:

- single source to single sink flow,
- multiple source and sink flow,
- circulation-only flow.

Derigs (1988, p 41–44) shows that an MCFP formulated in one context can be transformed to any other. The multiple source and sink flow statement is the most obviously general, as the other formulations require edges and possibly vertices to be added in order to capture some situations. The multiple source and sink formulation is adopted when formalising the MCFP in section 6.5.4. However, readers should note that the widely reviewed out-of-kilter algorithm (see p 90) is stated in terms of circulating flow.

Descriptively, the goal of MCFP is to find a (not necessarily unique) minimum cost flow configuration consistent with the network specification and given demand requirement. In the absence of an explicit demand requirement,[23] a least-cost maximum circulation flow configuration is generated. The commodity flow-rates become the network flows. The MCFP formulation requires that vertices be barred from accumulating the flow commodity, and that flow-rates are additive.[24] The optimisation techniques applied typically need costs to be linear with flow-rate. Non-trivial optimisation requires that redundant capacity exists within the network, such that useful routing decisions can be made.

Derigs (1988) shows that many other commonly encountered single-commodity flow-related network optimisation problems, including those concerning transportation, transshipment, assignment, maximum flow, and shortest path, are special cases of MCFP.

Interestingly, the network, as seen by MCFP, is now defined in terms of:

- its physical structure,
- the demand/supply profiles it faces — *ie*, its informational structure.

These two points demonstrate that networks are defined both by their structure and their operational state. This insight extends beyond the MCFP formulation.

[22] ′Fungible′ means able to be delivered in lieu.

[23] In this case the demand/supply vector $b = \mathbf{0}$.

[24] The additive assumption can be protected by setting an upper bound on flow capacity. By contrast, ′open-access′ transport networks will congest at some point, which means a small increase in traffic loading can result in massive decreases in traffic flows.

6.5.2 MCFP applied to network design evaluation

The MCFP optimisation problem just described can be used to support both operational decisions and design decisions. The operational context is self-evident. The ′network design problem′ or NDP comprises, equally, decisions relating to network architecture (the physical structure) and to demand management (the informational structure). Note that in this section, the form of the demanded commodity is left unspecified.

Operational problems and NDP share the same flow optimisation problem statement (*ie*, MCFP and extensions), but the NDP is conducted on a proposed rather than current network. Framing the NDP in terms of modification to a pre-existing network means that the design specification would require statements of the suggested network structure and/or of the anticipated commodity demand profile (by vertex). Hence, a design specification could include a network structure which requires new works, no change, or asset retirement, and a projected demand profile which increases, decreases, or redistributes demand, or remains unaltered.

It is salient to note that demand may be actively managed, at least to some degree, by pricing, compensation for restraint, request, and/or rationing. Indeed, explicit demand management may make better economic sense than adding to or maintaining parts of an existing infrastructure.

Design problems involving new works are often best served by a cost function which can capture both fixed and flow costs. Such an approach is known as the ′fixed charge problem′, and is discussed more fully on p 97. However, the use of nonlinear cost functions moves the problem out of the realm of MCFP, and brings a significant methodological overhead. The fixed costs required to extend network infrastructure are often treated as sunk, given that network fragments typically attract no economic opportunity cost beyond salvage.

Furthermore, a second and often antagonistic design goal is that of reliability, a concept which may be characterised by either vertex or edge connectivity and sometimes described by network engineers as ′diversity′.[25] Winter (1989) describes methods for searching networks (and sub-networks) for some minimum cost configuration whilst also maintaining a given number of disjoint paths between any two vertices. Note that diversity is of less value in the face of fault events which are not statistically independent.

It should be noted that some communication network problems (*eg*, airline travel) are best treated using optimal matching rather than flow network problems.

[25] Utilities networks are often operated with $n+1$ diversity, meaning that the network can continue to perform in the event of any *one* network element failing.

6.5.3 Mathematical approaches

Derigs (1988) demonstrates any MCFP can be framed in terms of:

- graph-theoretic optimal matching algorithms,
- network flow optimisation algorithms,
- linear programming (LP).

Derigs (1988) reports that linear programming methods are now favoured for reasons of numerical performance. It is, however, worth briefly reviewing the two alternative strategies.

Every flow network optimisation problem is potentially reducible to a graph-theoretic matching problem.[26] Algorithm development has favoured the network formulation, and, in practice, it is rare for flow network problems to be expressed in terms of optimal matching.

Network-based algorithms

Network-based MCFP algorithms include:

- minimum-cost-flow algorithm (Busacker and Gowen),
- out-of-kilter algorithm (usually attributed to Ford and Fulkerson).

The out-of-kilter algorithm, which is entirely general for the MCFP, was published in 1959.[27] Derigs (1988), Foulds (1981), Jensen and Barnes (1980), Smith (1982) provide descriptions of the algorithm. Smith also provides line-numbered BASIC and Pascal programs. Smith's Pascal code was translated into MATLAB 5.3 and the script is included in appendix D (see p 210).

The minimum-cost-flow algorithm is more restricted than the out-of-kilter algorithm (*eg*, minimum flow capacity must be zero) but is considerably simpler.[28] Foulds (1981), Gibbons (1985), Jensen and Barnes (1980) provide descriptions of this or similar algorithms.

All efficient flow network optimisation algorithms, including the out-of-kilter algorithm, build to a universal optimum using a sequence of rule-based decisions which classify as ′greedy′ and ′myopic′. It is noteworthy that decisions which involve only local structure will naturally result in a global optimum, and that the underlying algorithms run in polynomial time.[29]

[26] Matching problems can, in turn, be transformed into even more general mathematical factor problems.

[27] Derigs (1988) reports Yakovleva (1959) published first. The algorithm is usually attributed to L.R. Ford and D.R. Fulkerson, although it was published as Fulkerson (1961).

[28] Derigs (1988) reports the algorithm was published by Busacker and Gowen (1961).

[29] This issue of algorithmic time and space complexity is beyond the scope of this thesis.

Derigs (1988, p 31) reports that, until the early-1970s, the out-of-kilter algorithm and variants were considered more efficient than linear programming formulations for solving MCFP. However, from this point on, simplex-based methods gained ascendancy due to the use of graph-based rather than matrix-based data-structures, and improvements in starting and convergence strategies.

In the first instance, MCFP is framed in terms of integer programming (IP). However, as shown in the next section, linear programming (LP) techniques can be used to solve the problem. Both linear and integer programming form part of the larger discipline of mathematical programming.[30]

Linear programming

Linear programming is applied to constrained optimisation problems whereby both the objective function and constraint functions are linear with respect to the independent variables, and the constraints are expressed as either equalities or inequalities.

The simplex method, developed by George Dantzig in the late 1950s, is widely used to solve LP problems, although other powerful algorithms, including the interior point methods of Karmarka and of Barnes, have been introduced subsequently.

Briefly, the simplex method (in primal form) searches the feasible solution domain for the problem, represented by a bounded convex multi-dimensional polyhedron, by moving iteratively from one basic solution vector necessarily located on an extreme point to the next best adjacent basic solution vector, until the optimum is found. Multiple-optimal solutions result when the objective function finally aligns with two or more extreme points. The simplex method is very efficient for real-world problems, far more so than its algorithmic time-complexity would suggest. Minor refinements to the method allow certain operational pitfalls to be avoided.

Any correctly formulated LP problem can be suitably transformed to the simplex method 'standard form' using simple rules.[31] The most notable adaptation requires the introduction of slack variables to convert inequality constraint equations to equality constraint equations. Constraint equations need not necessarily 'bind', and redundant constraint equations are identified as part of the solution output. Problems which transpire to have no feasible solution or are unbounded are likely to be aphysical and in need of fundamental review.

The simplex method can be structured in two forms, the primal simplex method and the dual simplex method. Furthermore, every primal problem has an equally valid dual counterpart. The primal simplex method, described above, is the more

[30] In this context, 'programming' refers to the "determination of a plan of action that is in some sense optimal" (Casti 1996, p 185). The term pre-dates the advent of digital computing.

[31] Some authors use the term 'normal' form.

straightforward. The dual simplex method searches for a solution to the dual problem, and returns this result, which can be proved to be identical. Use of the dual form can have certain operational advantages.

Foulds (1981) provides an excellent summary of the simplex method and its refinements. His comments on post-optimisation analysis are particularly useful. Press *et al* (1993) give a clear description of the method. Derigs (1988), Nemhauser and Wolsey (1988) give succinct formal accounts of both the primal and dual formulations.

Alternative strategies

Excessive execution times may require the use of strategies which reduce the LP complexity of the problem without undermining the problem itself. One such strategy is Dantzig-Wolf decomposition, which breaks large LP problems into several sub-problems in situations where the vertex-edge incidence matrix exhibits 'block angular' structure. Foulds (1981) discusses this technique.

A range of discrete optimisation techniques exists outside of LP. One alternative, simulated annealing, is described by Press *et al* (1993). This and other strategies are discussed by Nemhauser and Wolsey (1988). Further research would be required to determine whether any of these alternative strategies have merit in the context of *deeco*.[32]

6.5.4 MCFP as a linear program

This section formally states the MCFP as a linear programming problem.

Formulation as an integer program

In the first instance, MCFP can be formulated as an integer program. The restriction to integer-valued variables is not as limiting as it may seem — any problem can be multiplied by some suitable scale factor (*eg*, 1000) and the results divided appropriately in order to obtain the required precision. The formulation proceeds as follows:

Let $G = (V, E)$ be a directed graph representing a given flow network. Let $v \in V$ represent individual vertices, and $e \in E$ represent individual edges.

Require that the directed graph G be specified using a vertex-edge incidence matrix

[32] Some proprietary LP routines are 'adaptive' and choose their method based on the metrics and structure of the problem submitted.

A with its rows representing vertices and its columns representing edges, such that:

$$a_{ve} = \begin{cases} 1 & \text{if } v = \text{head of } e \\ -1 & \text{if } v = \text{tail of } e \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

The network demand/supply (or capacity) function on the set of vertices is $b : V \to \mathbb{Z}$ ($\mathbb{Z}$ being the set of integers) where:

$$\begin{array}{ll} b_v < 0 & \text{indicates } v \text{ is a } \textit{source} \text{ with } -b_v \text{ flow-rate available} \\ b_v = 0 & \text{indicates } v \text{ is a normal vertex} \\ b_v > 0 & \text{indicates } v \text{ is a } \textit{sink} \text{ with } b_v \text{ flow-rate demanded} \end{array} \tag{6.5}$$

The network flow function on the set of edges is $x : E \to \mathbb{N}_0$ ($\mathbb{N}_0$ being the set of non-negative integers). Flow conservation at each vertex requires:

$$x(\delta^+(v)) \; - \; x(\delta^-(v)) \; = \; b_v \qquad \forall \; v \in V \tag{6.6}$$

where: $\delta^+(v)$ indicates the set containing incoming edges for vertex v
$\delta^-(v)$ indicates the set containing outgoing edges for vertex v

The network upper and lower flow bound functions on the set of edges are $l : E \to \mathbb{N}_0$ and $u : E \to \mathbb{N} \cup \{\infty\}$,[33] respectively, where:

$$l_e \leq x \leq u_e \qquad \forall \; e \in E \tag{6.7}$$

The network cost function on the set of edges is $c : E \to \mathbb{R}$ (more typically $\mathbb{R}_+$) ($\mathbb{R}_+$ being the set of positive real numbers) where:

c_e is the cost of transporting one unit from the tail to the head of edge e

Some authors define the now weighted graph as $G = (V, E, c)$.

The total cost for a given flow regime is the sum of all edge flow costs:

$$c(x) = \sum_{e \in E} c_e \, x_e \tag{6.8}$$

The MCFP is thus to find a set of feasible flows x which minimises the overall network cost $c(x)$. This problem can be expressed as an integer programming problem using matrix notation. Note that the vertex-edge incidence matrix A, which defines the graph's structure, also serves as the optimisation constraint matrix. Hence:

$$\begin{array}{ll} \text{min} & c^\top x \\ \text{subject to} & Ax = b \\ & l \leq x \leq u \\ & x \text{ integer valued} \end{array} \tag{6.9}$$

[33] Infinite capacity edges may be useful when transferring flow from sink to source.

where: A is the vertex-edge incidence matrix of size $|V| \times |E|$
b is the demand/supply vector of length $|V|$
c is the cost vector of length $|E|$
x is the flow vector of length $|E|$
l is the lower bound vector of length $|E|$
u is the upper bound vector of length $|E|$
$^\top$ indicates the transpose operator

Conversion to a linear program

Serendipitously, optimisation problems of this type involving directed graphs can be solved using linear programming rather than integer programming techniques. This is so because LP techniques applied to such problems can yield only integer-valued basic feasible solutions, as required.[34] Consequently MCFP can be restated as an equivalent linear programming problem, thereby allowing faster and less sensitive algorithms to be used.

The previous expression can be brought into simplex ′standard form′[35] by introducing slack variables $\tilde{x}$ in order to transform the inequality constraints. Here we also assume $l = \mathbf{0}$ which results in some loss of generality. Hence MCFP restated in simplex ′standard form′ becomes:

$$\begin{aligned} &\min && c^\top x \\ &\text{subject to} && \begin{bmatrix} A & \mathbf{0}_{|V|\times|E|} \\ I_{|E|} & I_{|E|} \end{bmatrix} \begin{bmatrix} x \\ \tilde{x} \end{bmatrix} = \begin{bmatrix} b \\ u \end{bmatrix} \\ & && x, \tilde{x} \leq \mathbf{0} \end{aligned} \tag{6.10}$$

where: $\tilde{x}$ is the newly introduced slack variables vector of length $|E|$
$\mathbf{0}_{|V|\times|E|}$ is the zero matrix of size $|V| \times |E|$
$I_{|E|}$ is the identity matrix of size $|E| \times |E|$

This problem statement can be solved directly using the simplex method.

Jensen and Barnes (1980, p 343) note that due to the optimum being a basic solution, flows in the nonbasic edges, which comprise a significant proportion, are either set at capacity or zero. In other words, flow distribution by edge tends to be highly po-

[34] The details are as follows. Matrix A is ′totally unimodular′ if integer-valued and if every nonsingular square submatrix of A has a determinant of ±1. The vertex-edge incident matrix of any directed graph can be shown to have such a property. Furthermore, it can be proved that an LP problem with a totally unimodular constraint matrix (in this case, the vertex-node incidence matrix A) and an integer-valued constraint vector (in this case, the demand/supply vector b) can only produce integer-valued basic feasible solutions. Hence LP techniques can be applied satisfactorily to what is fundamentally an IP problem. Refer to Derigs (1988, p 21–22, 29) for formal proofs.

[35] Foulds (1981, p 15–16) reviews the four steps that may be required.

larised. This state of affairs is more likely to be advantageous than disadvantageous when scheduling engineering plant.

Coding support

If programming in C/C++ (say for graph data-structure support) robust public domain source code for the simplex method is available,[36] or, alternatively, proprietary LP subroutines can be purchased.[37]

6.5.5 Network simplex method

Derigs (1988) notes that the network simplex method, which uses graphs rather than matrices for storage, is now generally preferred as the optimisation algorithm for MCFP. Derigs (1988, p 83–93) discusses the method in detail and formally states it as Procedure 7.2. Nemhauser and Wolsey (1988, p 76–82) also review the method, and note its speed is heavily dependent on the data-structure implementation used to represent the spanning tree.

Both sources cite a range of literature concerning the method's theory, refinement, and performance. In particular, Nemhauser and Wolsey (1988) cite Kennington and Helgason (1980) who give a program listing for use with large-scale data-sets.

The network simplex method uses the underlying network graph, rather than a tableau matrix to keep track of progress. It can be shown that a simplex basis B (obtained by setting selected structural and slack variables to zero) can be represented as a spanning tree T on the original graph, simply by treating B as a vertex-edge incident matrix in its own right.

The network simplex method proceeds broadly as follows. First determine a feasible network flow x, its associated spanning tree T (the current basis) and T's associated dual y (used next to calculate the reduced costs $\bar{c}_e$). Identify an edge e not in T such that its present flow and $\bar{c}_e$ meet certain conditions. If no e can be found, stop and report x, now optimum. Otherwise, form a directed circuit using $e \cup T$, search for a cost-augmenting flow, note edge f the weakest link, and revise x. Add e to and remove f from T. Calculate a new y and return to the second statement.

The network simplex method is reviewed primarily because it may prove useful when seeking efficiency improvements for *deeco*.

[36] For example, from Numerical Recipes (Press *et al* 1993, p 430–444). *deeco* makes use of Numerical Recipes code.

[37] For example, the MATLAB Optimization Toolbox `linprog` linear programming subroutine can be translated into C++ using the MATLAB Compiler, built using the system C++ compiler, and statically linked to a user's main program. `linprog` does not use the simplex method by default.

6.5.6 Extensions to the minimum cost flow problem

A number of extensions to the flow network optimisation problem as outlined have been developed. Extensions potentially relevant to *deeco* are summarised below, because a knowledge of these may help avoid or address future problems, and should also prove useful when selecting development paths for *deeco*.

Briefly, *deeco* faces three methodological issues. First, execution speed in the face of large data-sets (noting that most suggestions in this section will compound the problem). Second, optimisation across time-intervals is not possible. Third, linear costs may not always be appropriate. The background to these issues is discussed in detail elsewhere — the aim of the current treatment is simply to provide pointers to potential answers.

In terms of the literature, Winter (1989) surveys graph algorithms applicable to real-world network design and operation, and provides problem formulations and citations for most of the methods given here. Jensen and Barnes (1980) provide a less expansive coverage, dealing only with extensions that arise directly from MCFP.

Sub-network problems

Sub-network optimisation problems involve searching for optimum sub-network configurations, but the use of cheap routes outside the sub-network is considered to be valid. These problems, in general, are called Steiner problems, and most have only heuristic pseudo-solutions. The MCFP formulation cannot be modified to include this class of problem.

Convex costs

Cost functions which are convex on flow can be handled using two techniques.[38] First, a piecewise linear approximation can be applied using multiple edges, but this method introduces inaccuracies and results in a far larger graph to be solved. Smith (1982) describes this approach.[39] Second, Jensen and Barnes (1980) present algorithms capable of handling convex cost functions, but these are less efficient than LP methods. Jensen and Barnes also note that, in this case, optimal edge flows are less likely to be set at their flow bounds, in contrast with MCFP.[40]

[38] A function, over a given interval, is ′convex′ *iff* any line connecting any two points sits fully above the curve.

[39] Piecewise linear approximations can be readily implemented into *deeco* due to its object-oriented process module structure (see p 143).

[40] Jensen and Barnes (1980, p 347–350) demonstrate how an electrical circuit can be analysed by transforming the problem into a quadratic cost function flow network and optimising the transformed network. The reverse transformation can also apply.

Fixed-charge flow problems

Flow costs comprising fixed cost plus linear variable cost are considered more representative when appraising network design options involving new works. The fixed cost applies *only if* nonzero flow is assigned during optimisation.

This form of cost function requires mixed 0–1 programming, with substantially worse performance than MCFP.[41] Alternatively, approximate heuristic methods exist for some problem formulations. The sum of fixed costs can also be made subject to a constraint equation that represents the project's budget. Other forms of concave cost function are discussed in the literature. Collectively, these problems are sometimes referred to as network design problems or NDP, as discussed earlier.

Jensen and Barnes (1980), Marganti and Wong (1984), Winter (1989) review the fixed charge flow problem, while Nemhauser and Wolsey (1988, p 495–513) formulate it as a mathematical program.

Networks exhibiting edge loss or gain

Flow networks exhibiting edge gains can also be optimised using LP. A non-unity linear gain parameter means that commodity flow between the given pair of vertices will increase or decrease in transit as appropriate. Piecewise linear approximations using multiple edges can be constructed to model certain nonlinear gains. Even so-called generating loops are admissible if they meet modelling needs. This method may provide a useful way of representing losses in exergy networks, including transmission lines.

Jewell (1962) describes how to include gain parameters in the constraint matrix and then complete the LP formulation. Jewell terms such networks as ′process-flow′ networks. Jensen and Barnes (1980) also devote space to the topic.

Aggregated capacity constraints

Flow networks constrained by capacities applied to sets of edges rather than individual edges can be solved using polymatroid techniques. Lawler (1989) discusses this class of problem.

Probabilistic networks

Techniques have been developed to deal with networks whose attributes are defined using probability functions. One commonly encountered example is PERT (project evaluation and report technique) which supports stochastically-defined task times.

[41] Mixed 0–1 programming concerns mathematical programming problems where some variables belong to $\mathbb{R}_+$ and others to the binary set $\{0,1\}$.

Dantzig-Wolfe decomposition

As mentioned on p 92, certain network problems may be able to be partitioned, solved in smaller blocks, and reassembled. This technique should produce substantial reductions in execution time.

Fuzzy data compression

Another technique for reducing the problem size is to compress the time-series datasets prior to optimisation in a way that retains the essential structure. Groscurth and Kress (1998) describe the method.[42]

Parallelisation

If an algorithm or procedure can be parallelised — that is split into several chunks and run concurrently — then the opportunity presents to attack the problem on several CPUs at once. Parallel processing protocols are either tightly- or loosely-coupled, depending on whether or not they share common memory.

deeco could be not be parallised under its current design, because process state information is carried forward to the ensuing time-interval. If a parallel design were formulated, a number of implementation issues would need to be resolved.[43]

6.6 Additional topics in optimisation

6.6.1 Dynamic programming

The current implementation of *deeco* does not make use of dynamic programming (DP), however the method may have merit if *deeco* is developed further. Some understanding of DP is required in any case as DP is frequently encountered in the network optimisation literature.

Dynamic programming is a topic in mathematical programming. DP applies to serial systems — systems in which the current system state is dependent *only* on the previous state and completely *independent* of the way the system arrived at its previous state. Hence, DP uses time as a parameter and not as an independent variable. DP can be used to analyse multi-stage decision problems exhibiting the characteristics just described, and can be applied to both deterministic and stochastic processes.

Dynamic programming should not be confused with dynamic analysis — which applies to nonserial time-dependent systems, in which the state-space history of the

[42] Dr. Groscurth commented in late-1998 that he no longer favours this method for *deeco*.

[43] For example, threaded programming allows the programmer to implement tightly-coupled concurrency. However, it is useful to note that relatively few C++ libraries are guaranteed thread-safe, and no graph libraries to the writer's knowledge.

system is crucial. Dynamic analysis systems contain inertia and time lag, and may include feedback-based control.

DP provides a means of partitioning large problems by proposing a recursive relationship between sub-problems, which can then be solved in serial fashion.

Certain network routing problems are regularly tackled using dynamic programming. However this class of problem differs in structure from that resolved by the flow network optimisation methods given earlier. First, the entity being transported must be non-divisible, non-fungible, and destination-specific. Second, optimisation is defined from the viewpoint of the entity, not the network. Third, the optimisation result is a path rather than a flow regime — due to the fact route dividing is not possible.

Typical network applications include individuals planning least-cost or least-time multi-leg air travel, and data packages requiring least-delay routing through a data network. The technique proceeds as follows. Starting with the point of destination, all useable routes are followed upstream. Those for which less expensive alternatives have been identified are discarded. The process is repeated until all possible options back to the point of departure have been evaluated. Upon completion, the optimum path is the one that remains. Clearly, diversity is required in the network for the answer to be non-trivial.

Strategies have been developed for optimal entity routing which do not require the network state to be explicitly known, but instead rely on experiences relayed between so-called ′smart tribal agents′ (Ward 1998).

Foulds (1981) reviews dynamic programming. Yaged (1973) examines minimum cost routing in dynamic networks. Daellenbach, George, and McNickle (1983, p 232–237) frame a single route high voltage transmission line design problem in DP terms — this formulation is valid because transmission line costs are dominated by their non-flow-related component.

Discussions with Professor Richard Nowakowski[44] suggest it may be feasible to frame pro-active storage management within *deeco* as a DP problem. The strategy would be to work backward from the final time-interval, determining optimal storage as well as optimal routings as the procedure advanced. Intuitively, it would appear that the state-tranformation equations, which describe the intertemporal behaviour of *deeco* processes, would need to be invertible.

[44] During March 2000.

6.6.2 Multiple-objective optimisation

The methods discussed thus far involve only one objective function which represents a single flow cost. Certain general restrictions apply to such functions including the requirement that all costs be expressed in the same unit.

In many real-world situations, optimisation problems cannot be reduced to a single objective, as several non-synergistic and possibly non-commensurable objectives may exist. A number of formal techniques can assist in such situations. Some reframe the problem in terms of single-objective optimisation by:

- linearly weighting commensurable objectives (assuming commensurability),
- using second-string objectives as surrogate constraints,
- proceeding only with the principle objective followed by sensitivity analysis.

Others adopt true multi-objective techniques:

- minsum and minmax methodologies,
- use of multi-attribute usefulness functions,
- sequential solution methodologies.

In terms of the six techniques listed, *deeco* contains provision for the first, user-defined linear weightings. The Würzburg study (see p 160) made informal use of the second and third, in that these techniques were applied to the output from *deeco* scenarios, rather than being built into *deeco* itself. Post-output analysis of this type is entirely satisfactorily.

The remainder of the topics listed, which concern true multi-objective optimisation, are generally beyond the scope of the *deeco* model. Nonetheless, Daellenbach *et al* (1983) and Psarras, Capros, and Samouilidis (1990) discuss these and related techniques in more detail.

6.6.3 Stochastic simulation

Dealing numerically with model specification uncertainty is an area which falls outside the scope of this work, but warrants a brief mention. A number of stochastic simulation techniques have been developed to address this situation. One such method is Monte-Carlo analysis, however single scenario run-times for *deeco* in the order of 10′s of minutes effectively preclude its use.

deeco is deterministic in the sense that all aspects of the model specification must be defined explicitly, including the demand time-series and environmental data time-series. Notwithstanding, time-series could be modified statistically, to represent, say, an historical 100-return worst year, an average year, or some other chosen target. The issue of specifying data-sets which capture the near-future, in the face of environmental uncertainty, technological development, and changing preferences requires judgement on the part of the modeller.

6.6.4 Cost-benefit analysis

Cost-benefit analysis (CBA) is an evaluation methodology, rather than an optimisation technique. CBA requires direct and contingent valuations for all cost and benefit streams plus risk-adjusted discount rates for each class of stream. The methodology has little direct bearing on *deeco* although the output from *deeco* could well input to a CBA. CBA is also discussed on p 34.

6.7 Closure

This chapter provides specifications and techniques for dealing with a range of flow network problems, most of which were sourced from the operational research literature. The MCFP and extensions can be applied to operational problems and network design problems. NDP problems involving fixed costs are not catered for explicitly in the current implementation of *deeco*.

Getting to the core of these various optimisation algorithms is not strictly necessary. The main issue, as far as exergy network modelling is concerned, is to select a method which fulfils the modelling requirements and is efficient in terms of the metrics of the problems likely to be encountered, and then ensure that the method is implemented satisfactorily. □

Chapter 7

Description of the model

This chapter develops the model on which deeco is based.

7.1 Introduction

Chapters 5 and 6 introduced a number of ideas required to establish the model which underpins *deeco*— these ideas include:

- the suite of environment-dependent thermodynamic functions pertaining to exergy and ϵnergy,
- the two bulk commodity approximations,
- the multi-connection SSSF and NSSS plant abstractions,
- the logical exergy flow and logical plant abstractions,
- the demand plant and sourcing plant abstractions,
- the commodity-based flow network representation,
- the minimum cost flow network problem (MCFP) formulation.

An overview of the model was given in chapter 1. The current chapter revisits this material in more depth, except that the issue of representing plant input-output behaviour in mathematical terms is delayed until chapter 8.

7.2 A network view of exergy-services supply systems

This section outlines the concept of a generic exergy-services supply network by assembling such a network from its conceptual building blocks. It also emphasises the view that modelling can only ever capture an approximated subset of reality, and that in practice, a given model scenario is often forced by pragmatism to further compromise conceptual integrity.

7.2.1 Exergy-service

As noted (see p 4), this thesis is premised on the view that firms and households value exergy-services, rather than exergy use *per se*. This same concept is captured by the economic idea of 'derived demand' — with exergy-services being the item primarily sought and the exergy demand itself being secondary. Exergy-services are those services that arise, or could have arisen, from the active consumption of exergy.

Exergy-services, for a given system of interest, can be identified when one or more intensive variable, such as T, P, μ_i (temperature, pressure, species-specific chemical potential) or $\dot{x}$, z (horizontal velocity, altitude) are set and maintained at values which differ from that of the local dead-state. Common examples involve space heating (ΔT), materials processing ($\Delta\mu_i$), and the transport of goods or people (Δx, Δz).

All sustained exergy-service requires an ongoing input of exergy. In some cases, however, transient exergy services may be met from buffering effects alone. For example, a dwelling with a high mass, low diffusivity envelope will attenuate diurnal temperature variations, but the mean temperatures of both the house and ambient must equate.[1] Furthermore, with design modifications, solar gain effects can be used to raise the mean temperature of the house relative to ambient.

Therefore the provision of non-transient exergy-services requires suitable plant, and often but not always, some inbound exergy flow exhibiting nonzero opportunity cost. Thus firms and households often weigh capital investments in efficiency against avoided future exergy costs, as well as options costs. However, the end-user decision problem does not capture the full picture. For that, as discussed earlier, exergy-services supply systems need to be interpreted as a collection of patchworked flow networks.

7.2.2 Real networks

In both operational and developmental terms, a real network is taken as comprising the following — as determined by the problem at hand:

- the **physical plant** within the jurisdiction of either a single party, or grid-operators and grid participants,[2]
- the **demand profile** the network faces, by time and location,
- the **supply options** available to meet that demand, including import from gatewayed networks (*ie*, a patchworked network perspective),

[1] 'Thermal diffusivity' is given by: $D = k/\hat{c}$ where: D = thermal diffusivity [m^2/s], k = thermal conductivity [W/m K], $\hat{c}$ = heat capacity per unit volume [J/m^3 K].

[2] The concept of a 'grid' is discussed on p 114 and p 173.

- the minimum and maximum **capacities** of all plant, including transport plant (*eg*, line ratings[3]),
- information concerning the generalised **energetic performance** of all plant (*ie*, the input-output equation sets),
- the internal **storage management** policy in place (*eg*, for hydro-reservoirs),
- the **rationing policy** in the event of inability to deliver (*eg*, nodal pricing or priority supply),
- any **environmental conditions** of relevance (*eg*, weather, hydrology, thermal discharge restrictions, and institutional requirements).

Not all these aspects can be included within the model under discussion. In particular, rationing policy is not supported — indeed, if service demand cannot be met then the optimisation algorithm will terminate and return fail. But environmental constraints such as thermal discharge restrictions (*eg*, Huntly power station[4]) can be built into the model.[5]

Furthermore, not all the aspects listed are relevant to the model under discussion. The notion of a grid and grid-operator is not necessary for the model, but certainly may be relevant when dealing with real problems and actual policy recommendations.

7.2.3 The flow network abstraction

The most visible aspect of a flow network is its physical infrastructure. Conventionally, this infrastructure is viewed as technical plant linked by not necessarily loss-free interconnections carrying exergy flows. The exergy flows are predominantly oxidisable fuel flow or electric power — and interconnections therefore represent transhipment, piping, or wires. Topologically, the interconnections may range from zero diversity tree structures to high diversity nonplanar mesh structures, as indicated in figure 7.1.[6]

A description of the physical infrastructure provides only half the picture. A complete flow network specification requires knowledge of the supply options and demand levels by time and place — supply options comprise exergy capture, import, coproduction recovery, and storage drawdown, and the demand is for exergy-services. The end-use processes translate service demand into exergy requests, which are in

[3] Line ratings can be set by thermal limitations or by $n + 1$ operability considerations.

[4] Refer to footnote 12 (see p 112).

[5] *deeco* is able to restrict plant capacity on the basis of information contained within the environmental data time-series.

[6] Care is needed as 'diversity' can also be used to describe the way in which the electricity load profile looses volatility as one progresses back up an electricity grid.

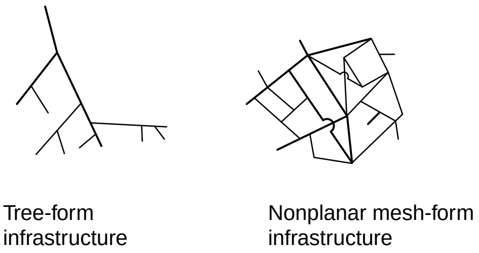


Figure 7.1 *Infrastructure topologies.*

turn forwarded upstream. Network specifications can be, and usually are, dynamic in time.

Exergy requests can be fully met only when the available supply options and required interconnections have sufficient capacity. If more than one set of supply options and/or interconnections can meet a given demand profile, then the network is said to exhibit redundancy under that specification. Diversity is based solely on issues of infrastructure, whereas redundancy factors in supply and demand.

Defining demand in terms of exergy-service rather than exergy *per se* vastly increases the scope of the system under consideration. The network now potentially encompasses housing stock, the transport fleet, and industrial processing. Not surprisingly, some rationalisation of the service view is required and a number of demands may be better represented by outright exergy requests.

An exergy-services network problem need not necessarily be single component — thus a particular network description can comprise two or more physically disjoint sub-networks.

It should be noted that not all problems can be approximated as networks — that is, made up of discrete connected elements. But fortunately interconnected exergy-services supply systems lend themselves to a network representation.

The linearity conditions imposed by the optimisation algorithm requires that changes in flow ϵnergy L_x (or, equally, flow exergy E_x) not be accompanied by significant changes in the bulk velocity of the stream υ, that is: $\Delta L_\mathrm{x} \gg \Delta\upsilon^2$. If this were to happen, then the linear relationship between currency and cost would break.

7.2.4 Thermodynamic description of the network

The underlying model is characterised in thermodynamic terms. As is often the case with thermodynamic analysis, pragmatic conclusions can be drawn without an in-depth knowledge of the processes under consideration. Thermodynamic analysis is taken to include the classical mechanics constructs of bulk momentum transfer and potential energy interchange — this being akin to the treatment used in exergy analysis.

The model is progressed by encapsulating thermodynamic processes as multi-connection plant and by requiring the interconnections joining various plant to be thermodynamically loss free — meaning that exergy cannot be lost or destroyed within an interconnection. On a thermodynamic level, the interconnections therefore represent system interfaces. And mathematically, the implied control volumes constitute the graph-theoretic dual of the underlying network structure. In addition, exergy flows re-circulating between two neighbouring plant need to be replaced by a single logical connection (see p 71 and p 148).

The most striking aspect of this new representation is that the model no longer explicitly captures physical distance. So while the interconnection topology remains congruent for both the physical and the thermodynamic representations, the spatial relationships become irrelevant for the latter.

A representative four plant network is shown in figure 7.2, using ideas introduced in chapter 5.

The accounted flow-dependent costs (irrespective of whether they are used in the optimisation or not) can be deemed to arise at the interfaces. These costs are then attributed to one or other of the two plant, or apportioned as appropriate.

The ′black-box′ behaviour of the plant themselves are encoded using input-output equation sets and related expressions, with further treatment of this issue left until chapter 8.

Transport losses can be captured by creating appropriate transport plant complete with exergy deficits. Reticulation losses include, for instance: static pressure drop and leakage in gas transmission, resistive losses and insulator leakage on AC and DC lines, and shunt losses (due to inductive coupling with their surroundings) on AC lines.

A connection interface is specified by statements covering exergy type and the exergy flow intensive attributes. Plant interconnections are only possible when both plant cut-points match, and interface integrity is maintained within *deeco* by enforcing this requirement.

The issue of resolving for the intensive state of the network prior to time-local optimisation is again deferred until chapter 8. But in order to be able to resolve the intensive state of the network certain modelling limitations regarding plant behaviour and plant interaction are required. These restrictions are not as onerous as might appear, as most of the plant likely to be encountered will comply by virtue of being actively controlled.

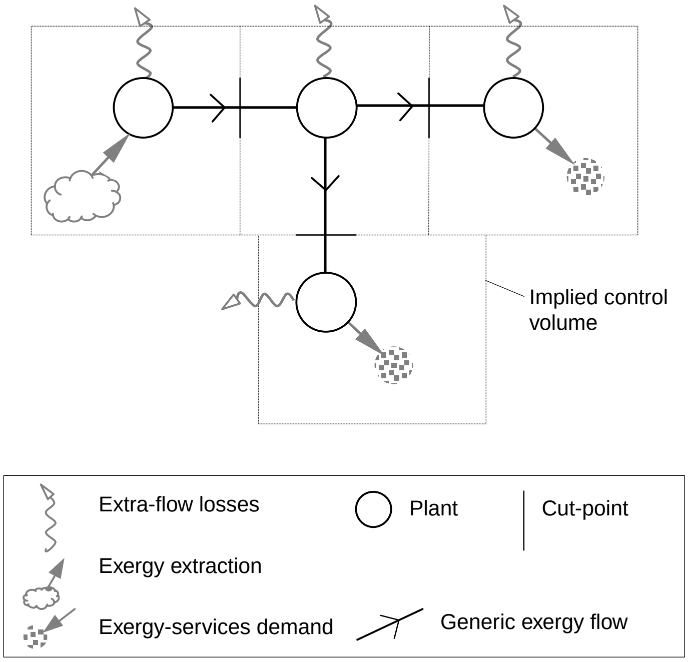


Figure 7.2 *A simple four plant network lacking storage.*

The operation of plant is often influenced by external factors, including the prevailing environmental conditions — for instance, windfarm output is a function of wind speed. The model allows for plant behaviour which is explicitly dependent on external influences.

Plant can be spatially aggregated if modelling precision is not compromised as a result. For example, several hundred similar domestic houses can be combined to give one suitably scaled ′mega-house′. Such aggregation should be leveraged wherever valid as it reduces the size of the computational task.

Model scope and network gateways

The notions of model jurisdiction and patchworked networks were introduced earlier (see p 15). Selecting an appropriate boundary for a given network problem is a matter for the modeller (and their client) with regard to the analytical purpose of the model. Network models are typically used for decision support, so that the network under analysis is often limited in extent or scope to that over which the modeller has authority, or at least some influence.

Consequently, models of the type under discussion require a clear statement of their domain, and this necessitates a clear identification of a model's boundary. This boundary can be both spatial and temporal — for instance, temporal boundaries are required if CO_2 sequestration (*ie*, forestry planting) is included. Unequivocal boundaries, in turn, enable cross-border transactions accounting. In many instances, models are defined ′in reverse′, using cross-border accounting to implicitly define boundaries and hence the scope of the network.

The network under consideration may well be fully stand-alone in the terms of the temporal boundary under consideration, but it is more likely to abut other networks with which it can transact exergy. In this case, gateway plant facilitate the import and export of exergy.

It follows, therefore, that an exergy-services supply system can be represented as a set of patchworked networks. Figure 7.3 depicts this situation, although the networks need not be planar as is shown. In addition, an individual network specification may comprise two or more disconnected components.

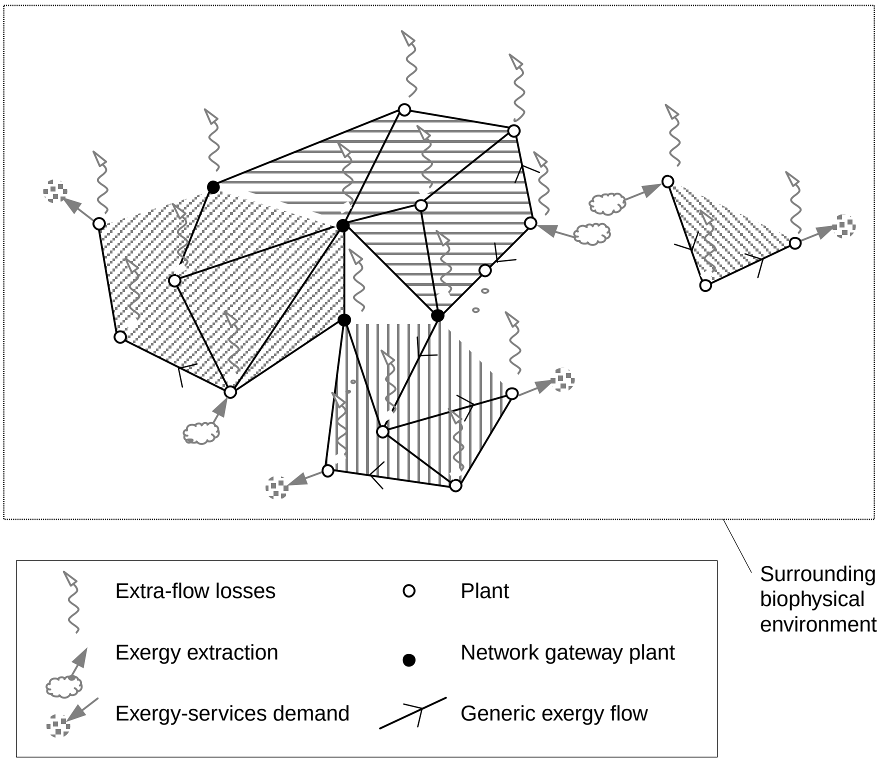


Figure 7.3 *The exergy-services supply system modelled as a set of not necessarily connected patchworked networks.*

7.2.5 Biophysical extraction

All exergy ultimately derives from the surrounding biophysical environment and is known in this thesis as ′biophysical exergy′. Biophysical exergy can be extracted via a number of resource pathways, including: petroleum deposits, river systems, windsheds, solar insolation, and biological harvest. Use of long recovery-time assimilative capacity can also be considered an extractive activity (see p 29). Primary $nergy is a (null, proper, or equivalent) subset of biophysical exergy.

In most cases, biophysical exergy has a high flow exergy quality factor Z but can vary markedly in terms of exergy flux (*ie*, current density) and temporal availability. Furthermore, in many instances, biophysical exergy is unpriced and free from resource royalties.

7.2.6 Plant efficiency, exergy storage, and demand management

Plant efficiency gains at any point in a supply chain can be interpreted as virtual supply to that point. This applies to passive as well as active plant (see p 170). For instance, thermal insulation, which assists exergy retention, will displace upstream exergy requests.[7] Similarly, heat transfer augmentation, which reduces exergy destruction, will again displace upstream exergy requests, for a given situation.[8]

As noted previously, exergy storage within the network can be used to buffer the short-term temporal mismatch of supply and demand, and over quite long periods. For instance, thermal storage can support seasonal variation, *ie*, half-yearly. Exergy storage used in this context can be interpreted as virtual system capacity, in that it reduces the sum of conversion infrastructure required to meet maximum demand. Moreover, the addition of storage generally displaces depletable sourcing with renewable sourcing, given good integration. Demand shifting can be interpreted as virtual storage. And export/import can also be viewed as a form of storage, albeit with less management autonomy than internal storage.

In general, plant efficiency improvement and storage require investment in physical equipment. As a result, efficiency investment is often assessed using localised financial appraisal, although difficult to monetorize secondary benefits may influence the decision to invest. In addition, improved end-use efficiency may increase exergy-service demand, particularly where the short-run marginal cost of providing the exergy-service falls as a result (Pearce 1998).

Demand management in its various forms, by contrast, requires no change in infrastructure, but instead seeks to alter the prevailing demand profiles.

7.2.7 Sustainability and integration

As discussed earlier, the generic ESSS sustainability debate can be partitioned into two aspects: resource depletion — in the widest sense — and local environmental impact. However, some additional comments are relevant here.

[7] This concept is sometimes described colloquially as the production of ′negawatts′.

[8] *ecco*, a forerunner of *deeco*, contained virtual supply as a process category.

Issues relating to population growth are not traversed in this thesis, primarily because the topic is complex and its inclusion would add little. If need be, population-driven demand increase can be factored in by re-scaling demand profiles over time.

The issue of $nergy supply security has historically been framed in terms of exergy delivery rather than exergy-services provision — this treatment unfortunately tends to draw attention away from customer-domain efficiency opportunities, on-site exergy capture, and the strategic use of storage (including high-mass architecture), as well as gains arising from better system integration. Furthermore, aggregate exergy-services demand reduction constitutes a legitimate response provided such reduction is not accompanied by deprivation. Experience shows that, in many cases, the opposite is true — reduction can often be accompanied by positive rather than negative social side-effects.

The conventional response to the problem of mid-term supply security has been to redouble efforts to source biophysical exergy, particularly those options based on non-distributed technologies. But new options, based on improved integration are slowly gaining credence. Such opportunities include: better integrated hydro-reservoir management, improved heat-exchanger network (HEN) design, the use of co-generation, district heating with or without seasonal storage, passive solar architecture, and better time synchronisation of demand and supply. Nonetheless these options are generally implemented on an *ad hoc* basis — arguably a more structured and encompassing approach, as advocated in this thesis, would yield better results.

But systematic integration requires a set of cost functions to be articulated and agreed, consistent with the overarching system goal. In terms of ESSS sustainability, minimising net CO_2-e emissions provides a reasonable optimisation objective, particularly given current technologies. For many networks, net CO_2-e emissions correlate well with other environmental impacts and with depletable resource use.

7.2.8 Storage management and intertemporal optimisation

Storage management is important. The most straightforward strategy is to replenish storage plant using zero optimisation cost sourcing whenever surplus output is available and then use this inventory to help meet demand whenever this option contributes to the lowest cost flow regime. This strategy, which lacks ′insight′, is unlikely to yield the optimum use of storage capacity in situations when future demand and supply profiles can be anticipated. An improvement would be to explicitly optimise storage across time-intervals using dynamic programming (DP) techniques, which may then require nonzero-cost replenishment and time local suboptimal use.[9]

7.2.9 Decision support

The issue of decision support is covered in detail elsewhere (see p 11). The comments which follow are intended to add to that discussion.

Flow network optimisation modelling can be used in two ways. First, it can be applied to an existing network in order to identify best practice in context of the selected goal. Second, it can be applied to a proposed network modification to test which technologies, interconnections, and/or demand profiles would be best included and encouraged, again in context of the selected goal.

Optimisation modelling of the type under consideration is better suited to provide interpreted information for system policy formulation, than provide uninterpreted input for operational control. The reason is that optimisation techniques can never capture the full complexity of a problem, and other factors not built into the model invariably need consideration in an operational context.[10]

Optimisation modelling may reveal that some proposed network modification provides substantial sustainability improvements and yet faces amortised monetary costs similar to that of the chosen reference case. Selecting the least destructive regime in this situation would be described as ′win–win′ in today's jargon — noting that win–win options fall in the lower right quadrant of the trade-off diagram (see p 18). Furthermore, the trade-off between any two regimes which weigh improved sustainability against increased monetary cost can be quantified — and thereby produce a shadow price for the impact. But care is needed when interpreting such results, as the concept of incrementality need not necessarily apply to ESSS networks — a small change in circumstance may lead to a substantially different optimal flow regime and result in a shadow price function which is markedly discontinuous.

7.2.10 Internalisation

There is reasonable prospect that certain negative environmental externalities arising from ESSS may be internalised in the near future, using mechanisms such as taxation, feebates,[11] mandatory compensation, or tradable impact entitlements. If and when such measures are applied, ESSS networks will face a new monetary cost

[9] Scott and Read (1995) and Yang and Read (1999) describe intertemporal hydro-reservoir optimisation using dynamic programming techniques.

[10] The wholesale electricity nodal pricing system used in New Zealand follows this strategy. Although the SPD flow-weighted demander cost minimisation model used by Transpower outputs an optimal generator schedule, the grid operator still dispatches power stations manually. So in effect, the electricity system operates using administered control, albeit informed by least-cost analysis.

[11] Feebates are revenue-neutral tax/rebate schemes.

environment, and the sustainability/monetary cost trade-off should become further predisposed toward sustainability.

Alternatively, network plant may be required to operate within some mandatory impact entitlement envelope. In New Zealand, this type of operational constraint can be included as RMA resource consent conditions,[12] in addition to other statutory and common law duties that may apply.

7.2.11 Competition and synergy within the network

Optimisation studies indicate that groups of technologies under a given demand profile may either compete with, coexist with, or complement one other. Competition occurs where, say, two ostensibly similar technologies can supply a given service in full, but subtle variations in external conditions over time mean that the optimisation routine toggles between the two — with the result that both technologies are under-utilised and neither performs well. Coexistence indicates the absence of this type of dynamic. And complementation describes the opposite situation.

Bruckner *et al* (1997, p 1009) report several examples of technology competition — for instance, heat-pumps without co-generation and *vice versa* give notable gains for sustainability at little or no increase in monetary cost — but when both technologies are applied, the gains are only trivially better whilst the capital cost is additive. These effects are illustrated on figure 1.3 (see p 18) which shows a pattern of (approximately) reducing returns as the technology mix is broadened. Limiting the available capacity of competing technologies should assist the overall trade-off, and this may provide a better option than rejecting one technology altogether. On the other hand, most plant shows diminishing output-normalised costs with scale. The main observation is that only situation-specific numerical modelling can give fully robust answers.

However, public policy imperatives may require more sweeping conclusions. In which case, a second-best option is to use optimisation modelling to analyse typical situations — which are then generalised, albeit at some risk.

7.2.12 A network integration index

Network optimisation — applied in an operational context — can be used to benchmark the prevailing management regime in terms of some flow-dependent cost criteria of interest. Furthermore, the following ratio would produce a useful indication

[12] For example, the 1000 $\mathrm{MW_e}$ Huntly coal/gas-fired thermal power-station cannot raise the temperature of the Waikato River to above 25°C at specified points. Hot dry conditions mean that station output must be severely reduced, often to less than 50%.

of network management performance:

$$\text{integration index (for given cost criteria)} \quad = \quad \frac{\text{actual performance}}{\text{best practice performance}} \qquad (7.1)$$

This index could be applied to situations where net CO_2-e reporting is mandatory — in which case, the reporting could provide the data on actual performance, whilst integration modelling would provide the estimate of best practice.

7.2.13 An operation definition for ′energy efficiency′

Network modifications — with location-aggregate service demand held static — which result in an overall reduction in depletable energy use are arguably more closely aligned with public policy usage of the term ′improved energy efficiency′ than with any other definition. For instance, a recent report from the Parliamentary Commissioner for the Environment (2000, p 1) states: "energy efficiency means any change in energy use that results in an increase in the net benefits per unit of energy." Notwithstanding, the report contains no advice as to how this definition may be quantified or even meaningfully applied. This thesis argues that integration modelling using *deeco* (or an equivalent environment) can overcome such deficiencies.

7.2.14 Modelling under *deeco*

The steps involved in constructing and interpreting a network integration model using *deeco* are as follows:

- state and refine the research question,
- identify the scope of the network under consideration,
- confirm that all model restrictions (see p 151) are acceptable,
- define the network, including the demand for exergy-service time-series and the environmental data time-series,
- obtain the required cost information,
- select the flow-linear cost to be optimised,
- prepare and check the data-set,
- run the scenario,
- interpret the results.

7.3 Non-model issues

The ESSS flow network optimisation model presents useful qualitative insights in addition to providing numerical results. The model cogently places a number of disparate policy concepts — including efficiency, renewable energy, demand management, storage use, waste recovery, and merit-order dispatch — under a common conceptual framework.

But two issues of practical merit not captured by the flow network model also deserve attention. The first is that of collective interest management, and the second is the question of structural paradigms.

7.3.1 Grid-mediated supply and demand

While, for most networks, the bulk of the infrastructure in terms of economic value is operated under individual jurisdiction, significant sections are managed to serve the collective interest of those they interconnect (or could potentially interconnect). These sections, as subsets of the overall system, are termed ′reticulation grids′ or simply ′grids′ and their purpose is to transport a particular exergy commodity, such as natural gas or electric power. The collective interest may be vested in either public (*ie*, institutional) or club (*ie*, participant) management regimes. The trend in New Zealand has been to favour privatised rule-making.

Grid-mediated exergy supply and demand usually offers significant advantages. The comments that follow assume that the supply options available are substantially inhomogeneous in time and location in terms of their monetary cost, environmental impact, and operational attributes, and furthermore, that network-effect market failure and price discrimination have limited influence.[13] Potential aggregate benefits include: reduced capital requirements, cheaper primary inputs, decreased environmental impact, improved commodity quality, improved supply reliability, and greater flexibility in terms of demand plant siting. The downside for participants is a possible loss of flexibility as they will need to comply with grid connection standards. Some participants may also choose to place their plant under grid operator management, typically in return for reduced tariffs.[14] How best to identify and value the benefits and disbenefits listed above is a research field in its own right and beyond the scope of this thesis.

[13] ′Network-effect market failure′ arises where a given network locally saturates and the prevailing market regime (*eg*, opportunity cost-based nodal pricing) provides for monopoly pricing (Harvey and Hogan 2000 — noting that their discussion is limited to transmission constraints).

[14] Ripple-controlled domestic water heating is an example, whereby the local grid operator can unilaterally switch off electric hot water cylinders by final transformer block for customers who elect ′controllable′ tariffs. Ripple control is not common outside New Zealand.

7.3.2 Structural paradigms

A number of analysts have suggested that the monolithic source/reticulate paradigm that accords with most contemporary exergy supply systems will be increasingly displaced by a distributed resource (DR) paradigm. This latter paradigm favours the following: distributed generation, distributed storage, multi-product plant, proactive demand management, and lighter bi-directional grids. (Fuge *et al* 2000, Lenssen and Flavin 1996, Patterson 1999, von Weizäcker, Lovins, and Lovins 1997).

Such a shift could be quite rapid and may lead to the widespread stranding of existing assets together with their embodied impacts. Drivers for change include, on the one hand, emerging technologies with falling costs, and on the other, an existing system facing declining short-run and long-run security and substantial reinvestment requirements. Exergy sector liberalisation programmes appear to be assisting the trend toward distributed resources, despite the presence of in-built policy hostility toward this paradigm.[15]

The optimisation model presented in this thesis is *independent* of the network paradigm, and can be used to test the comparative merits of the two paradigms just discussed, or any other variant.

7.3.3 The design of incentive structures for participants

Network integration can be difficult to achieve in practice. This is because the issues requiring resolution typically span a number of participants, each with divergent aspirations. The design of an incentive regime which aligns these various aspirations with the drivers for network integration, and moreover, is not susceptible to abuse, is particularly difficult. Still, this thesis argues that detailed systems modelling is necessary for the development of robust incentive regimes, irrespective of whether operational accountability is overseen by system participants or public institutions.

7.4 Closure

One of the strengths of *deeco* is that the conceptual model on which it is based also provides a cogent qualitative description. Furthermore, the act of building a valid numerical model forces the underlying conceptual model to become better articulated. Nonetheless, robust conclusions can only arise from numerical modelling, because network-based systems generally exhibit counter-intuitive behaviour on a scale sufficient to rule out ′first principle′ inferences, particularly in situations where at least some of the plant reach their capacity constraints. □

[15] Parliamentary Commissioner for the Environment (2000) reviews some of the hostilities faced by distributed resources in New Zealand.

Chapter 8

deeco : dynamic energy, emissions, and cost optimization

This chapter examines the UNIX*-based application program deeco.*

8.1 Introduction

The conceptual building blocks which underpin *deeco* were reviewed in previous chapters. This chapter concentrates on the application itself, but is not intended to double as software documentation. Readers are also directed to the two references listed on p 20, which together describe much of the formalism behind *deeco* and to the current software documentation listed on p 155.

deeco is a UNIX-based application program used to identify integration opportunities for existing and planned exergy-service supply systems with the aim of improving their sustainability. *deeco* also accounts for and reports fixed and variable monetary costs, but cannot be used to explicitly optimise for these. *deeco* was originally structured to deal with municipal exergy system problems, and may require adaptation on the part of the modeller if applied to other contexts.

The bulk of this chapter describes the application itself, whilst the final part reviews two published studies undertaken using *deeco*.

Note on terminology

As shown in table 8.1, some of the terminology adopted in *this* chapter aligns with current *deeco* documentation and thus varies from that used elsewhere in this thesis. The mathematical notation used is consistent with that in the rest of this work, unless specifically noted. The symbols L and Λ are used for ϵnergy, whereas E (suitably scripted) is used within the *deeco* documentation for the same concept.

deeco **documentation**	**Elsewhere in this work**
flow attributes	cut-point intensive values
net-enthalpy flow	logical heat flow
network process	transport plant
process	plant
chemical energy flow	mass-flow fuel
differential performance index	marginal εnergetic perfomance index
efficiency	εnergic efficiency
energy	flow εnergy
fixed costs	flow-independent costs
heat flow	actual heat flow
information flow	data-flow
input/output relation	input-output equation
link	edge
node	vertex
non-renewable primary energy	depletable exergy
NSEE	process surroundings
supply path	supply chain
time-variant	interval-dependent
variable costs	flow-dependent costs

Table 8.1 *Terminology differences between the deeco documentation and this thesis. The terms in bold are used locally in this chapter.*

Several terms require comment at the outset. First, those of a general nature. *State* is used in its mathematical sense, in which case the context should indicate which set of variables under consideration. *Heat* may be accorded a logical interpretation which is more encompassing than its strict technical definition (see p 148). *Flow* refers to generic exergy flow unless the context indicates otherwise. The terms *flo* and *return* indicate outward and inward heat transfer media circulation respectively, as viewed from the exergy supplying process. *Parameters* are used to particularise equations — in the case of *deeco* the parameters are part of the data-set and cannot be changed during a model run, yet may be interval-dependent and given as time-series. *Compile-time* and *run-time* describe the points at which a particular application program is built and executed, respectively.

Second, terms relating to graphs.[1] *Static* indicates that the structure, once formed, is invariant. *Bipartite* graphs contain vertices that can be partitioned into two subsets, each without internal linking. *Simple* graphs have only one edge connecting any two vertices. Conversely, *multi*-graphs have multiple edges. *Directed* graphs, or *digraphs*, contain directed edges. *Nonplanar* graphs cannot be drawn on a sphere without intersecting edges, whereas *planar* graphs can. *Cyclic* graphs contain directed cycles

[1] Graph theory is covered in chapter 6.

whereby a directed path starts and finishes at the same vertex, conversely *acyclic* graphs are free from directed cycles. The term *parallel* is used in its directed sense.

Use of 'rate': because *deeco* is framed in terms of discrete time-intervals, the qualifier 'rate', as in flow-rate and cost-rate, is not required. In a more general context this qualifier should be included.

8.2 Development history

deeco was preceded by several numerical optimisation projects undertaken or led by Dr. Helmuth Groscurth.[2] These culminated in the two DOS-based applications: *ecco*: energy, cost, and carbon dioxide optimization, and *ecco*-solar, both written in Borland Pascal. The *ecco*-solar extension was written by Dr. Thomas Bruckner as part of his diploma thesis. *ecco* and *ecco*-solar are reported in Groscurth, Bruckner, and Kümmel (1993).

deeco was developed and coded by Dr. Bruckner in collaboration with Professor Reiner Kümmel and Dr. Groscurth, as part of his PhD research whilst at the Institute for Theoretical Physics, University of Würzburg, Germany.

The Institute purchased a state-of-the-art Hewlett-Packard (HP) workstation for the project in mid-1992. At the same time, Dr. Bruckner began work on the underlying model framework, given the acronym NEMESS: network model of energy-services supply systems. The model framework was finalised in late-1994 and is published as Groscurth *et al* (1995).[3] Coding proper commenced and the application was completed in late-1995, as version 0.5. Dr. Bruckner then undertook the Würzburg study (see p 160) which concluded in early-1996 and is published as Bruckner *et al* (1997).

The writer's involvement with *deeco* began in October 1997 and is outlined in chapter 2. The *deeco* source code was ported to an Intel-compatible platform and testing was complete by November 1998.

During this time, Dr. Lindenberger, also at the Institute for Theoretical Physics, ran *deeco* to evaluate solar district heating options using the original HP workstation (see p 161). Dr. Lindenberger wrote new process modules in order to model the solar-driven heat storage sub-system. This study is published as Lindenberger, Bruckner, Groscurth, and Kümmel (2000).

[2] Personal affiliations are listed on p iii and p 20.

[3] Groscurth *et al* (1995) does not contain a description of the thermal sub-network algorithm, but details of this algorithm are available in Bruckner (1997, translation).

Subsequently, Dr. Lindenberger compiled *deeco* on an Intel-compatible personal computer using the writer's modifications. Dr. Bruckner is now a faculty member at the Institute for Energy Engineering, Technical University of Berlin, and is continuing work on *deeco*. As can be seen, a modest development community has started to take shape.

8.3 Overview of *deeco*

This section describes *deeco*. Readers should note that more than one reading may be required, as the concepts involved are difficult to present in a fully linear fashion.

8.3.1 Introduction

deeco is built around the twin concepts of the dynamical exergy process and the flow network — with both discussed in depth in chapters 5, 6, and 7. *deeco* uses abstract data-types to represent engineering plant and plant interconnections within the model domain, this being achieved by constructing discrete processes and containing them within a graph structure.[4] Each process manages its own affairs, namely, its engineering state and behaviour. Further, processes routinely exchange information with their neighbours and with their abstracted surroundings. Information transfer between neighbours is particularly important, and the data-channels involved can be collectively visualised as a virtual directed data-channel graph. The three principal algorithms within *deeco* can then be interpreted as operating on various data-flow subgraphs of this data-channel graph. *deeco* is also fully deterministic, in the sense that no stochastic procedures are involved. *deeco* accounts for exergy flow and process performance using ϵnergy rather than exergy.[5]

8.3.2 Procedure

deeco runs from a command line interface (CLI) and hence is not event-driven. Upon execution *deeco* reads in a user-provided scenario definition from a suitably structured ASCII input data file-set, and, on completion, writes out an ASCII output data file-set for subsequent interpretation. Each individual model run is known as a scenario.

The key steps required to optimally integrate a given scenario are shown in figure 8.1. *deeco* is structured around three principal algorithms:

- TSN — the thermal sub-network algorithm,

[4] This chapter makes use of a number of computer science terms and concepts. Illingworth (1996) is recommended to readers requiring a specialist computer science dictionary.

[5] This arrangement is discussed further in footnote 6 (see p 4).

- TLFO — the time-local flow optimisation algorithm,
- SRE — the storage replacement and export algorithm.

These algorithms, also indicated on the right side of figure 8.1, are discussed further in section 8.3.6. The five key steps that make up each iteration are as follows:

- update process characterisations, using external information,
- finalise flow attributes, using the TSN algorithm,
- time-locally flow optimise the resulting network, using the TLFO algorithm,
- replenish storage, and export, in that order, using the SRE algorithm,
- report process-specific information back to the process surroundings,
- increment the interval index, and loop.

By default, *deeco* iterates through 8760 hourly time-intervals to give a time-horizon of one year. As indicated, the optimisation is time-local, but storage inventory is always replenished as circumstances allow and carried forward minus losses to the next interval.

8.3.3 Model abstraction

Model abstraction is the task of identifying the key aspects of a problem and representing these in a manner which can, in turn, be translated into a numerical program.

Network representation

Figure 8.2 shows a network schematic of a simple municipal problem.[6] The individual processes are interposed with balance points — used to split and combine flows and also to enforce flow type and attribute integrity. The exergy flow interconnections themselves are indicated as connections. The thermal sub-network, shown as an overlay, uses single-connection logical exergy flows to replace the dual connection *flo* and *return* pipework which exists in reality (see p 71). The flow attributes (*ie*, the intensive values) of the heat transfer media contained in this sub-network must be resolved prior to each optimisation step. The problem as shown does not contain storage, so storage replenishment need not be undertaken after each optimisation step — nevertheless, it may prove advantageous to add storage in appropriate form — as part of a long-term network design strategy.[7]

[6] The network structure depicted is similar to that in `testdata`, one of the data-sets used to test the *deeco* port.

[7] Thermal storage is accounted using thermomechanical nonflow exergy Ξ_{x}, noting that, in general, both the pressure and temperature of the storage medium may differ from ambient. Kotas (1995, p 51–55) discusses this concept but uses the term 'physical non-flow exergy' Ξ_{ph} instead.

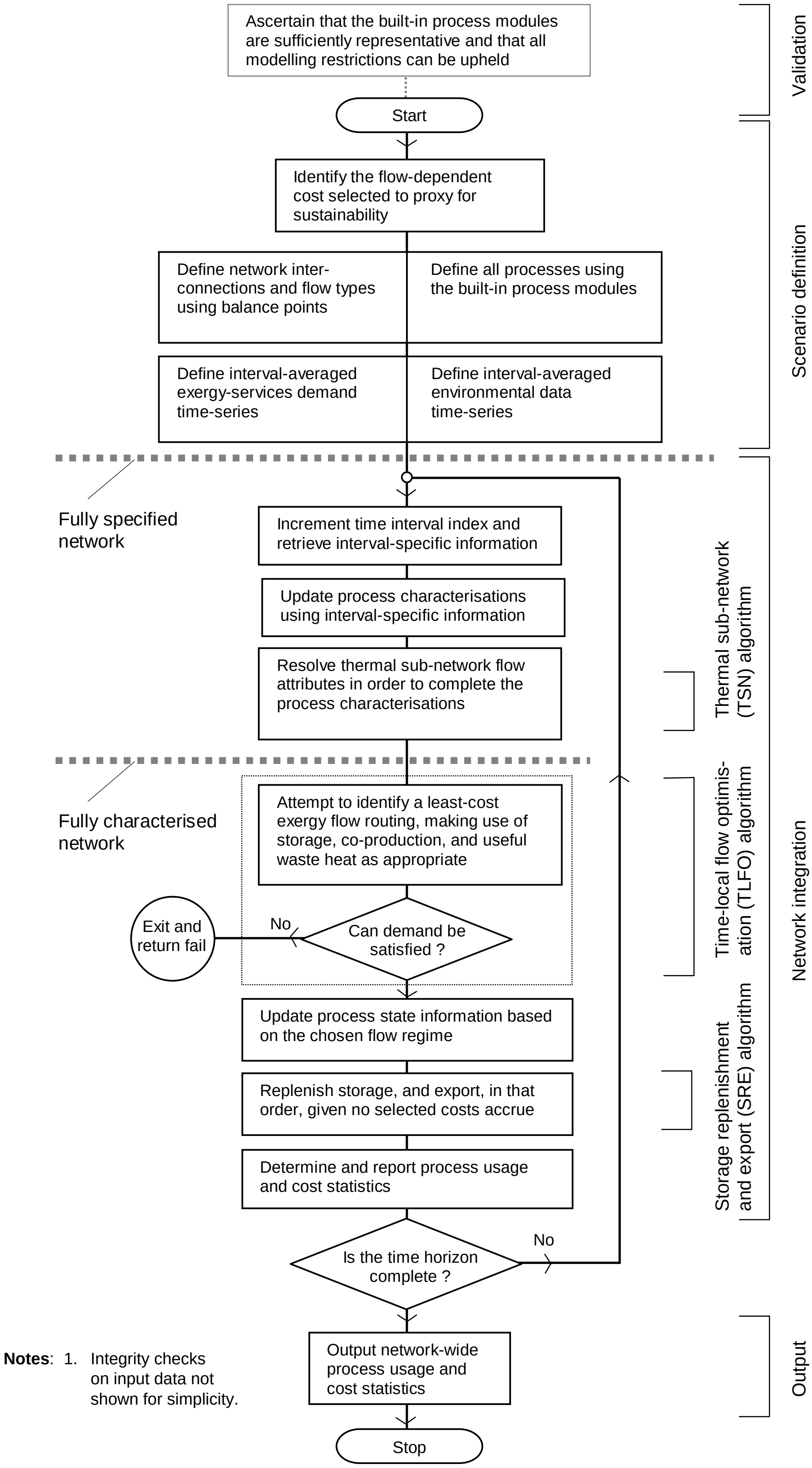
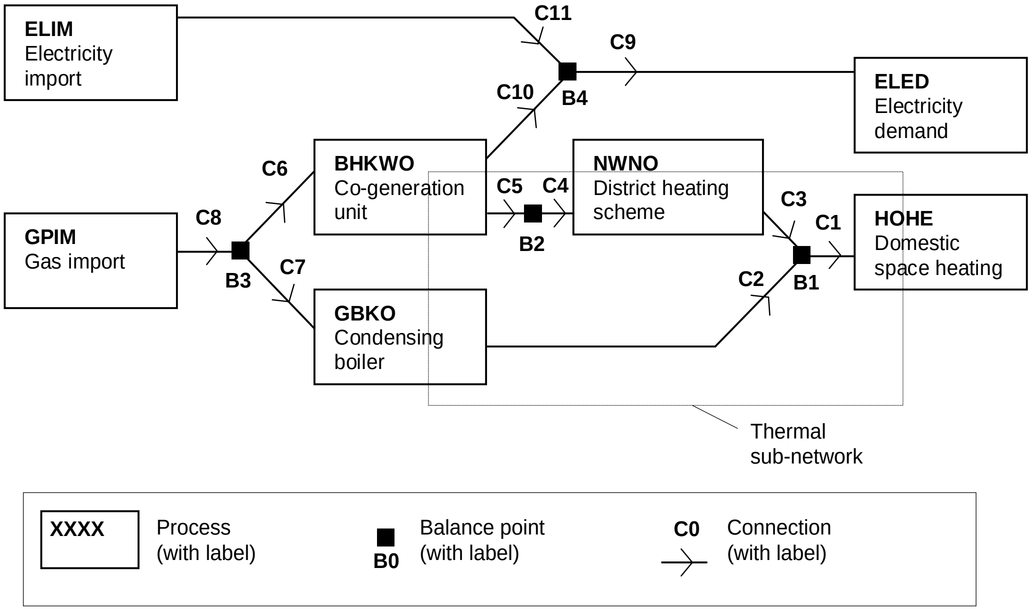

Ascertain that the built-in process modules are sufficiently representative and that all modelling restrictions can be upheld
Start
Identify the flow-dependent cost selected to proxy for sustainability
Define network inter-connections and flow types using balance points
Define all processes using the built-in process modules
Define interval-averaged exergy-services demand time-series
Define interval-averaged environmental data time-series
Fully specified network
Increment time interval index and retrieve interval-specific information
Update process characterisations using interval-specific information
Resolve thermal sub-network flow attributes in order to complete the process characterisations
Fully characterised network
Attempt to identify a least-cost exergy flow routing, making use of storage, co-production, and useful waste heat as appropriate
Can demand be satisfied ?
No
Exit and return fail
Update process state information based on the chosen flow regime
Replenish storage, and export, in that order, given no selected costs accrue
Determine and report process usage and cost statistics
Is the time horizon complete ?
No
Output network-wide process usage and cost statistics
Stop
Notes: 1. Integrity checks on input data not shown for simplicity.
Validation
Scenario definition
Network integration
Output
Thermal sub-network (TSN) algorithm
Time-local flow optimisation (TLFO) algorithm
Storage replenishment and export (SRE) algorithm

Figure 8.1 *Flow-chart outlining deeco.*



Notes:
1. The labelling notation is similar to that required by the ASCII input data file-set format.
2. *deeco* is written such that balance point **B4** is not required, instead a global balance point or 'pool' serving all electricity reticulation is built into the application.
3. Similarly, the electricity connections required to operate the various processes are not shown, these derive automatically from the global electricity balance point when these processes are instantiated.
4. The thermal sub-network indicated encloses a group of processes exchanging heat, in this case via forced convective transfer. Hence, the single connection logical exergy flows shown replace the *flo* and *return* pipework which exist in reality.
5. A gas distribution process is not needed as long as reticulation compressor exergy use is low, in which case gas reticulation costs should be assigned to the import and conversion processes as appropriate.

Figure 8.2 *Network schematic showing a simple municipal problem with co-generation and district heating, but lacking storage. There is no requirement for the network to be planar.*

This network structure is embedded in a graph, with processes and balance points mapped to vertices, and the exergy flow interconnections mapped to directed edges. This graph becomes the ′complete exergy interconnections digraph′ or complete EID, and, in its most general form, classifies as static, bipartite, simple, directed, nonplanar, and cyclic — but subject to certain restrictions described shortly. Processes and balance points comprise the bipartite subsets.

Directed data-channels also link the processes, but only where exergy interconnections also exist. These directed data-channels are mostly present in single antiparallel form with respect to the underlying exergy interconnections. In addition, processes exchanging heat require up to two directed data-channels, whose orientation depends on circumstances to be discussed in due course. Further, the interposed balance points naturally branch these channels to match the configuration of the underlying exergy interconnections — this general arrangement being shown as figure 8.3. Figure 8.4 (see p 126) also depicts the inter-process data-channels under discussion, marked as horizontals.

Directed data-channels also link processes and their surroundings, but these channels are not included in the present discussion. Figure 8.5 (see p 137) shows such channels, marked as verticals.

The directed data-channels carry different types of information between processes, depending on the algorithm in operation. The TSN algorithm transmits *flo* and *return* attribute information, whereas the TLFO algorithm transmits exergy requests. The directed data-channels also carry environmental data and demand information *to* the process from the environmental data and demand time-series, and cost information *from* the process to the network cost registers.

The collection of directed data-channels which link processes is again embedded in a graph, known as the ′complete data-channel multi-digraph′ or complete DCM. This graph maps to the vertices of the complete EID, and so inherits many of its properties. The three principal algorithms operate on data-flow subgraphs based on the complete DCM.

At this point, a certain modelling ′shortcut′ is introduced (which also benefits users of *deeco*). As indicated in note 2 on figure 8.2, electricity reticulation is modelled as a ′pool′ rather than as a grid (see p 70). As a consequence, individual electricity interconnections within the network problem do not require specific definition — instead *deeco* generates the required exergy interconnections and balance point (singular) automatically. This feature reduces the amount of effort required to produce a scenario definition, given that most processes demand auxiliary electricity in order to operate. The use of an electricity pool is reasonable for municipal supply, but

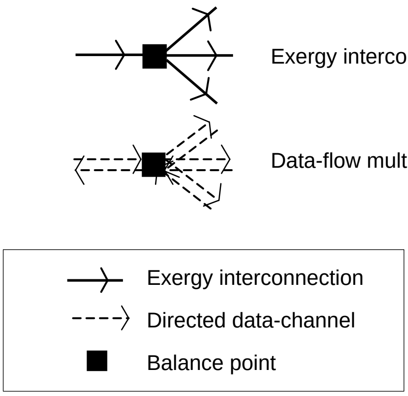

Exergy interconnections digraph vertex
Data-flow multi-digraph vertex
Exergy interconnection
Directed data-channel
Balance point

Figure 8.3 *An exergy flow splitting balance point and associated directed data-channels. The arrangement shown indicates the general case for data transfer, but implies nothing as to purpose, information type, or mechanism of carriage.*

it does prevent electricity reticulation losses from being easily modelled.[8] Hence, the process connectivity definitions necessarily exclude the exergy flow type **El** and contain only the types: **Mech**, **H**, and **Fuel** (the exergy flow types are summarised in table 8.4, p 129). The subgraph which results from the removal of the **El** edges and their global balance point is described here as the ′restricted exergy interconnections digraph′ or restricted EID (the *deeco* documentation uses the term ′non-electrical ESS (energy supply system)′).

Cyclic structures, which manifest as directed cycles on the exergy interconnections digraph, are formed whenever output from a process is fed back upstream of the process itself. For example, a thermal generation plant may draw electricity from the pool for essential purposes (*eg*, lubrication pumps) but may also generate into the same pool. Cyclic structures are discussed at some length elsewhere (see p 70).

The TLFO algorithm is fully able to operate on digraphs containing directed cycles (one variant of this algorithm is formulated in terms of circulating flow). Therefore, this algorithm can (and must) operate on the complete exergy interconnection digraph, or more precisely, the anti-parallel data-flow digraph which arises from the complete EID.

By contrast, the TSN algorithm (which immediately precedes the TLFO algorithm) requires that the restricted EID on which it operates be acyclic. Therefore, *deeco* enforces a general restriction on undirected (which includes directed) cyclic structures arising from **Mech**, **H**, and **Fuel** interconnections or any combination of these

[8] The work-around to this problem is to define a new exergy flow type **Elc**, so named to indicate ′connected electricity′. Alternatively, the existing **Mech** type could be used as an interim measure to transport electrical work.

— this then allows the TSN algorithm to operate satisfactorily on the restricted exergy interconnections digraph. This restriction is not particularly onerous as such situations are generally avoided in practice in any case.[9]

Intra-network storage adds more complexity. Storage processes, if present, toggle between two roles — they are either available to supply exergy flow given adequate inventory during the TLFO algorithm, or to be replenished during the SRE algorithm. This means that all replenishment exergy flow interconnections need to be disabled during flow optimisation. The complete EID, with these edges and any stranded balance points removed, is known as the 'optimisation exergy interconnections digraph' or optimisation EID. This graph also has an associated 'optimisation data-channel multi-digraph'. In general, the optimisation EID and the restricted EID only partially intersect.[10]

Supply chains

A contiguous sequence of exergy flows which links an exergy source process to a demand process is known as a 'supply chain'. The definition is not required to implement *deeco*, but can prove useful when discussing scenario findings. In particular, the notion of a 'marginal supply chain' which results from an trivial increment in demand can be informative — but noting that the network load/routing behaviour can be discontinuous if the scenario definition contains certain nonlinearities — such as those introduced via process cut-in thresholds and maximum capacities, for example.

Generalised process instance

A generalised process instance is shown in figure 8.4, complete with exergy interconnections, directed data-channels, process module-specific information, process instance-specific information, and process interval-specific information. This process can only be linked to other processes via balance points.

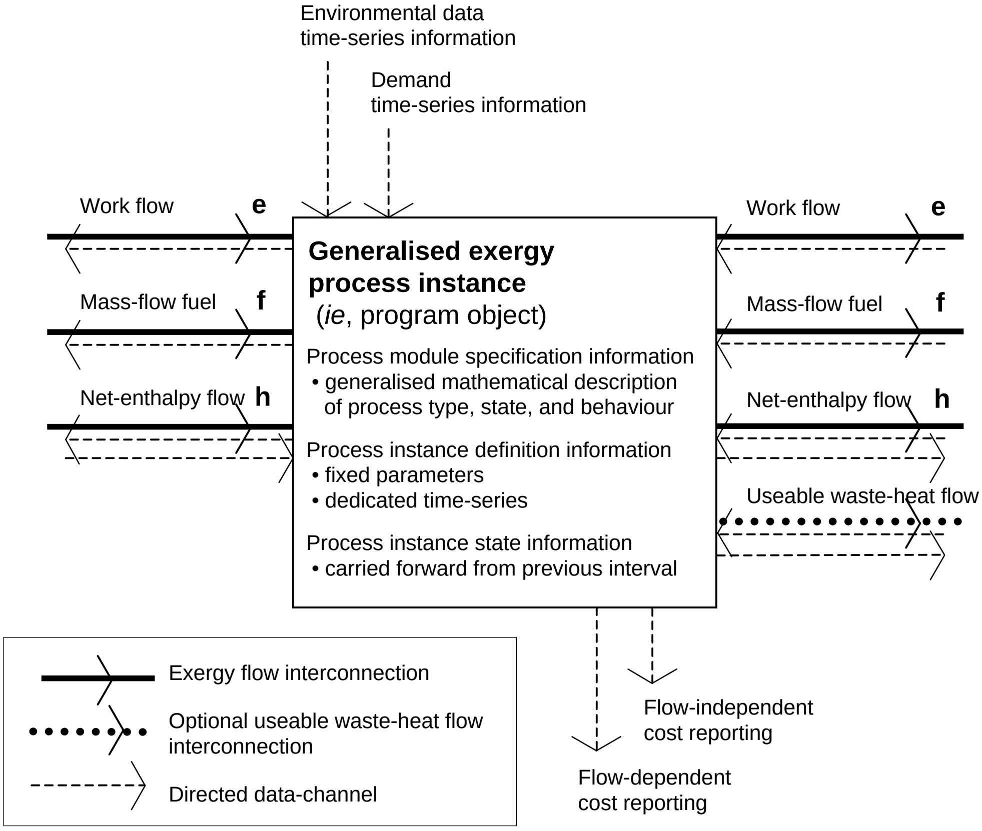
Processes and balance points

deeco supports six categories of process as given in table 8.2. Each category contains one or more built-in process module, each based on a corresponding process type.[11] Instances of process modules are used to construct the network infrastructure in the model domain. As noted, *deeco* requires that processes be interposed

[9] It is possible that the cyclic restriction need only apply to **H** fuel types.

[10] Restated in mathematical terms: optimisation EID $\otimes$ restricted EID $\neq \emptyset$, where: $\otimes$ is the symmetric difference operator. $A \otimes B$ is defined by $\{x : x \in (A \setminus B) \vee x \in (B \setminus A)\}$, where: $\setminus$ is the set subtraction operator and $\vee$ the or operator.

[11] The term 'module' is not used here in the computer science sense of separately compiled software units, although a process module may well be such.


Environmental data
time-series information
Demand
time-series information
Work flow
e
Mass-flow fuel
f
Net-enthalpy flow
h
Work flow
e
Mass-flow fuel
f
Net-enthalpy flow
h
Useable waste-heat flow
Generalised exergy process instance
(ie, program object)
Process module specification information
• generalised mathematical description of process type, state, and behaviour
Process instance definition information
• fixed parameters
• dedicated time-series
Process instance state information
• carried forward from previous interval
Flow-independent cost reporting
Flow-dependent cost reporting
Exergy flow interconnection
Optional useable waste-heat flow interconnection
Directed data-channel

Notes: 1. Net-enthalpy flow includes actual heat flow and natural and forced convective heat transfer.
2. Work flow includes electrical and mechanical exergy.
3. Process modules may be specified with some, all, or multiple exergy streams of the types shown.
4. The various directed data-channels are used at different steps during each iteration and sometimes for more than one purpose.
5. Useable waste-heat flow could more properly be labelled 'useable waste-net-enthalpy flow'.

Figure 8.4 *A generalised deeco process instance as it exists within the model domain.*

by balance points. Balance points are used programmatically to enforce exergy flow type and flow attribute (*ie*, intensive values) integrity, as well as to facilitate stream splitting and stream joining. The balance points themselves are inert, and allow mass, exergy, and data-flow to pass unaltered and unhindered. Exergy loss, be it, in reality, through destruction or leakage,[12] can take place only *within* the processes themselves.

Process category	Symbol	Comment
demand processes	**d**	demand is generally specified as exergy-service
conversion processes	**c**	transform fuel type and sometimes increase quality
network processes	**n**	transport exergy with loss, also includes heat-pumps
import/export processes	**p**	act as gateways in patchworked networks
storage processes	**s**	buffer exergy flows of all types
collector processes	**o**	harvest renewable exergy including solar thermal
virtual supply processes	**v**	not supported in the current version of *deeco*

Table 8.2 *Process categories used in deeco.*

The balance points themselves represent process boundaries and, as an added convenience, can be used to split and join flows given no exergy loss occurs. If reticulation does involve nontrivial losses and/or nontrivial exergy demand (*eg*, to run gas pipeline compressors), and/or nontrivial costs (*eg*, capital expenditure) then a dedicated exergy consuming and/or destroying network process (**n**) would be required.

Furthermore, the balance points ensure that the incoming and outgoing exergy flow types and flow attributes match exactly. The flow attributes are the intensive values contained in a particular flow. Any mismatch would result in exergy destruction (disallowed) or exergy fabrication (aphysical).

Four process categories require special mention because these ′create′ and ′annihilate′ the exergy flows within the model — such a statement is, of course, only true within the domain of the model and does not apply in any wider sense.[13] Figure 5.5 (see p 72) depicts these abstract sourcing and demand processes.

Creation process categories comprise import (**p**) and collection (**o**). Each import process requires details of all relevant flow-dependent and flow-independent costs embodied in the imported exergy stream, which can be given as interval-dependent. Each collector process requires suitable environmental time-series data.

[12] Leakage refers to the plant containment — all leakage is ultimately ′destroyed′ within the plant control region, necessarily placed some distance from the plant wall.

[13] Readers may, if they wish, construct an overarching source and overarching sink to explain process-local sources and sinks, but this is not mathematically necessary.

Annihilation process categories comprise demand (**d**) and export (**p**). Each demand process requires suitable demand time-series data, and can also call on environmental time-series data. After time-local optimisation, any zero-selected-cost exergy flow which cannot be used to replenish storage is exported. Exported exergy generates negative flow-dependent costs equal to that of the production it displaces in the gatewayed network. In reality, demand processes lacking waste-heat recovery also destroy the exergy supplied to them.

Storage processes (**s**) can be regarded as creation processes during optimisation, and annihilation processes during recharge. The use of storage processes requires some explicit storage management policy — the regime adopted in *deeco* is non-anticipatory: discharge as appropriate and replenish wherever the selected cost is zero.

The earlier *ecco* model also supported a virtual supply processes (**v**), by which passive measures, including insulation, could be selected to offset actual exergy requirements. This process category is not currently implemented in *deeco*, although it was included within the NEMESS framework. Experience with *ecco* shows that virtual supply (*eg*, building insulation) would almost always be selected, and therefore it is acceptable to incorporate all sensible passive measures into the demand processes from the outset. The robustness of this approach could be evaluated, if necessary, by comparing two scenarios differing only in demand process definition.

Certain demand, conversion, and other processes are also able to produce usable ′waste-heat′ which can be reused locally, *eg*, for space heating. In this case, the now logical process contains two vertices (there is no internal balance point) — the first represents the normal process itself, and the second creates any available and demanded waste-heat flow. This separation is necessary to prevent the formation of cyclic structures within the restricted exergy interconnections digraph.[14]

Some process definitions take interval-specific parameters which need to be supplied as dedicated time-series. For example, an electricity import/export process (of category **p**) may require interval-dependent conversion efficiencies and/or flow-dependent costs so as to capture changes in marginal generation in the abutting network.

Further information relating to process categories can be found in Groscurth *et al* (1995, p 946–955). Normally, users of *deeco* need not concern themselves with specification of process modules beyond that required to verify applicability to the plant being modelled. Yet some insight into the way process modules arise, and process instances operate, may be valuable.

[14] The *deeco* documentation notes that waste-heat flow should be described as waste-net-enthalpy flow but that this term has limited intuitive appeal.

Exergy flows and flow accounting

deeco supports three categories of exergy flow as shown in table 8.3. Synthetic carriers could include H_2(g,l) (hydrogen) and NH_4OH(aq) (dissolved ammonia).

Exergy flow category	**Symbol**	**Comment**
work flow	**e**	includes electrical and mechanical flow exergy
mass-flow fuel	**f**	includes depletables, biofuels, and synthetic carriers
net-enthalpy flow	**h**	defines a thermal sub-network

Table 8.3 *Exergy flow categories used in deeco.*

The **e**, **f**, and **h** categories map onto the terms E_W, $E_{\mathrm{x|t}}$, and E_Q in the flow exergy equation, given in eqns (5.1) – (5.12) (see p 57).[15] Net-enthalpy flow **h** can be taken as the collective logical form of natural and forced convective heat transfer, conductive heat transfer, and radiative heat transfer.

Within *deeco*, the exergy flows and process performance are accounted using flow ϵnergy. But equally *deeco* could have been structured using flow exergy, as this quantity is also conserved within the edges of the flow network and linear with cost. Notwithstanding, framing plant performance in terms of exergetic efficiency would be difficult — engineering equipment is almost universally documented using ϵnergetic efficiency. But irrespective of whether the commodity flows are accounted using E or L, the underlying commodity flow model remains best viewed as generic exergy.

The *deeco* documentation does not use the categories given in table 8.3, but instead uses the more detailed exergy flow type classification shown in table 8.4.

Exergy flow type	**Symbol**	**Category**	**Treatment**	**Comment**
electricity	**El**	**e**	work flow	uses a global balance point
mechanical work	**Mech**	**e**	work flow	uses normal balance points
actual heat transfer	**Q**	**h**	net-enthalpy flow	treated as special case of **H**
logical heat transfer	**H**	**h**	net-enthalpy flow	modelling restrictions apply
mass-flow fuel	**Fuel**	**f**	enthalpy flow	based on enthalpy of devaluation
depletable exergy flow	**P**	cost	enthalpy flow	due to importing

Table 8.4 *Exergy flow types used in deeco.*[15]

[15] The symbol $E_{\mathrm{x|t}}$ is shorthand for E_{x} or E_{t} (see box on p 56).

[15] The net-enthalpy flow types given in Bruckner (1997, translation), fig 2.3 should not be confused

Electrical exergy (**El**) interconnections need not be specified for the reasons given in section 8.3.3. Net-enthalpy flow (**H**) (defined in eqn (8.5), see p 146) provides a single interconnection representation for forced convection. Further, actual heat transfer (**Q**), which comprises conduction and radiation, is treated programmatically as a special case of net-enthalpy flow. Useable waste-heat flow also classifies as net-enthalpy flow. Depletable exergy flow (**P**) is essentially a flow-dependent cost and is not used to interconnect processes.

The flow types also possess flow attributes. Most flow attributes are given directly as part of the process specifications, but the attributes of some net-enthalpy flows may need to be resolved at each iteration using the TSN algorithm.

Network connectivity is established by linking processes and balance points, and defined as ASCII input data. Balance points and processes contain information as to their flow type, so *deeco* checks the integrity of this data upon commencement.

Processes and process interconnections in programming terms

In computer science terms, a process module class is instantiated by a particular process definition and this creates a fully specified process object within the model domain. All process objects are constructed at the commencement of the program and persist for the duration of the program. Once formed, a process object will update its state and modify its behaviour as its circumstances — represented by the data passed to it — change. In line with the principle of encapsulation, the mathematical representation of process behaviour is held in member functions, and, similarly, process state information is held by data members. The individual process objects and their interconnection details are retained in a graph container object which also bundles useful traversal and search procedures. Process instances are clearly dynamic objects, but the graph object, once fully formed, remains static.

The directed data-channels which, collectively and partially, make up the various algorithm-specific data-flow digraphs, are not held in separate graph container objects. Rather, they are artefacts of the traversal routines undertaken by the algorithms themselves. But the notion of algorithm-specific data-flow digraphs, although adding a further layer of abstraction, is nonetheless a useful concept when describing the prosecution of the model.

Process and network specification

A network is said to be 'fully specified' when all of its processes are fully specified and all their interconnections are defined using balance points. This occurs at run-time, after the process instances and graph container have been constructed. A network

with the exergy flow types described here.

is said to be ′fully characterised′ when all of its processes are fully characterised. This occurs just prior to flow optimisation and takes place for each iteration.

Each process is treated as a ′black box′ in that its inner workings need not be understood. Instead, it is only required that the process can be represented mathematically, subject to certain restrictions imposed by the algorithms contained within *deeco*. The full specification of a process in the model domain is undertaken in two stages.

First, a mathematical description, in parameterised form, of the process type is finalised, using performance data based on engineering design formula or sourced from data-books and/or equipment manufacturers. This description is then coded, and with other programmatic information, used to build a new process module. The task of specifying process modules requires C++ programming skills and access to the source code, as well as engineering knowledge.

Second, the process module, duly built, is instantiated using a process definition supplied as ASCII input data. The process instance acquires a full set of parameter values, which may be fixed or interval-specific, as required by the process module. At this point, the process instance now has a full description of its initial state and its interval-dependent behaviour, including cost creation. This behaviour, however, can still be modified by external influences and, in certain circumstances, neighbouring processes. The task of defining processes requires information about the plant being modelled, and some technical knowledge.

The network structure is established by defining the connectivity between balance points and processes, again supplied as ASCII input data.

Further discussion on the mathematical representation of processes is deferred until section 8.4. Readers may wish to consult figure 8.6 (see p 144) ahead of time, although some of the concepts shown are yet to be introduced.

Process and network characterisation

The network must be fully characterised prior to each execution of the TLFO algorithm, which requires that all processes be fully characterised. Process characterisation means that the flow attributes of all flows connected to the process (including *flo* and *return* media), the mathematical relationship between all entering flows, exiting flows, and internal storage (if present), and mathematical statements describing all costs consequent on these flows, are fully resolved.

Process characterisation is a four-stage exercise, following on from the TLFO algorithm. First, the process state is updated in light of the identified flow regime. Second, the SRE algorithm is run, and the process state again updated as appropriate. Third, the mathematical descriptions governing the state and behaviour of the process are recast, using (as required) internal interval-specific parameters, external information pertaining to the new interval, and internal information carried forward from the previous iteration interval. And fourth, the *flo* and *return* attributes of heat-exchanging processes are resolved — a two pass procedure using information transfer between neighbouring processes, and undertaken by the TSN algorithm. These steps are depicted in figure 8.1.

8.3.4 Scenario definition

Each model run constitutes a scenario — noting that *deeco* can be instructed to perform several model runs from a single execution. A scenario definition requires the information below be supplied via the ASCII input data file-set:

- process definitions,
- interconnection details,
- environmental data and demand time-series,
- choice of cost function.

The data-entry format for *deeco* is somewhat clumsy, but not onerous compared with the time required to collect time-series data, identify process types, and set the process parameters. Data entry details fall outside the scope of this thesis, refer instead to the application documentation.

Process definitions

The information required for a process definition depends on the complexity of the process type and the way in which the process module has been structured. This information is supplied using fixed and interval-dependent parameters. At the risk of oversimplification (a more robust description is given in section 8.4) these parameters input into equations which describe (as required) the following:

- process ϵnergetic efficiency and related performance indexes,
- process cut-in threshold and process maximum output,
- process load-independent maintenance exergy flow type and demand.

In addition, these parameters also input into equations which generate the following flow-independent and flow-dependent costs (as appropriate and as required):

- fixed (*ie*, amortised over its design life) and variable monetary costs,
- embodied and flow-dependent net CO_2-e emissions,
- embodied and flow-dependent depletable exergy use,
- embodied and flow-dependent sulphur, NO_x, and particulate emissions.

Details of supported process modules, their underlying theory, and their parameter lists are given in the application documentation (mostly in German) cited on p 155.

Interconnection details

Interconnections details are specified by defining the connections between processes and balance points.

Environmental data and demand time-series

Under *deeco*, the definition of environment includes both natural and socio-economic factors. The environmental data time-series data requirements are governed by the particular processes contained within the scenario. For example, a windfarm process requires wind-speed information by time-interval. In addition, socio-economic and operational effects can be included in the environmental data time-series, by partially or fully disabling processes on an interval-by-interval basis or for the entire time-horizon. For instance, public appeals for conservation could be factored in, if the results could be anticipated. Similarly individual processes could be made unavailable, perhaps for maintenance or to simulate economic gaming.[16] Demand time-series data requirements are similarly set by the particular demand processes present.

The environmental data time-series, the demand time-series, the flow-independent costs, and the flow-dependent costs, are collectively described as the ′process surroundings′. The *deeco* documentation uses the term ′natural and socio-economic environment (NSEE)′ for the same concept.

Statistical techniques can be used to regenerate environmental data and demand time-series, if the underlying data-set cannot be obtained. However, important fine-grain structure may be lost, so this technique should only be used as a last resort.

Choice of cost function

deeco supports the cost functions shown in table 8.5. Costs subject to optimisation are required to be linear on exergy flow, or be amenable to the approximation given in section 8.5 (see p 143).

Depletable exergy use (**PE**) arising from imported electricity is calculated using a conversion factor which may be interval-dependent. The net CO_2-e content (**CO2AE**) is similarly calculated using an emissions factor.[17]

The developers of *deeco* suggest use of LHV (lower heating value) to report import-related depletable exergy use. This is only acceptable when used with fuels having

[16] ′Gaming′ involves operators *purposefully* withholding capacity in the expectation that the ensuing price rise will more than compensate for decreased sales.

[17] McMullan (1999, p 7) provides a typical value of 1 GJ generated electricity = 181 kg CO_2. Bruckner *et al* (1997, p 1007) use 1 GJ of wholesale electricity = 198 kg CO_2 and 205 kg CO_2-e. Note: 1 PJ/year = 278 GWh/year = 31.5^{-1} GW.

Cost function	Variable	Flow-dependent unit	Amortised flow-independent unit	Comment
sulphur emissions	**SO2**	kg/J	kg/s	—
NO_x emissions	**NOX**	kg/J	kg/s	—
particulate emissions	**STAUB**	kg/J	kg/s	for example, PM_{10}
ghg emissions	**CO2AE**	kg/J	kg/s	reported as CO_2-e
depletable exergy use	**PE**	J/J	J/s	LHV accounting suggested
monetary cost	—	\$/J	\$/s	optimisation not supported

Table 8.5 *Cost functions supported in deeco. The amortised flow-independent costs are based on the peak-load faced by a process during the scenario and linearly depreciated over its accounting lifetime.*

an exergy quality factor near unity, otherwise flow exergy-based accounting is indicated.[18] Unfortunately national energy reporting protocols usually use LHV to account.

deeco also supports the addition of user-defined flow-independent and flow-dependent costs beyond those built into the application.

Cost calculation

Flow-dependent (or variable) costs arise from the exergy flows chosen by the TLFO algorithm, and flow-independent (or fixed) costs arise from the process being invoked at some point within the time-horizon. The flow-independent costs for a particular process are calculated by multiplying its load-specific flow-independent costs by its maximum output within the time-horizon, and are reported in both amortised and lifetime form. The amortised form is determined by linearly amortising the total flow-independent costs over the defined useful lifetime of the process — known in accounting as the ′straight-line method′.[19] The contributions from the individual processes are totalled to provide a network-wide tally at the end of a model run.

A process which operates for just one time-interval will necessarily report its (amortised or lifetime) capital and embodied costs for the selected level of duty. This is usually, but not necessarily, wasteful and undesirable — in which case the modeller should disable the under-utilised processes and rerun the model. As a result the flow-dependent costs will invariably rise, and the trade-off between these and the decreased flow-independent costs can be examined.

[18] The flow exergy quality factor Z is given in eqn (5.17) (see p 65). LHV is discussed on p 176.

[19] It is not necessarily appropriate to use the ′economic life′ of the process, as applied to financial appraisal calculations, for this purpose. The economic life is determined by profit expectations and can be quite short.

deeco can only optimise for one flow-dependent cost per scenario. The normal procedure is to run several scenarios using a suite of selected costs, in the hope that one solution exhibits costs that regress well and show worthwhile improvement. Decision support issues are discussed at length elsewhere (see p 11 and p 111).

Monetary cost

deeco tallies the monetary costs for a given scenario and reports these upon completion as just described. Monetary cost reporting allows financial trade-offs between different scenarios to be more readily assessed. Flow-dependent monetary cost can be used as an explicit optimisation goal, but this practice is not recommended for two reasons. First, the strategy inverts the optimisation problem by relegating improved sustainability from the key goal to merely one of several problem constraints. Second, fixed monetary costs typically dominate variable monetary costs, so little useful information will result — however, the reverse typically applies to sustainability costs, thereby rendering valuable results.

8.3.5 Scenario output and data visualisation

Scenario output

deeco outputs the following information:

- the flow ϵnergy each process outputs or inputs,
- system horizon-averaged fixed and variable monetary costs,
- system horizon-averaged embodied and flow-dependent net CO_2-e emissions,
- system horizon-averaged embodied and flow-dependent depletable exergy use,
- system horizon-averaged embodied and flow-dependent sulphur, NO_x, and particulate emissions.

Note that only one cost per scenario can be selected for optimisation purposes. However, provision exists for some linearly weighted combination of costs to be used.

Breadboard representation

A breadboard representation provides an useful way of depicting complicated non-planar structures.[20] Hence this method is suited to *deeco* as demonstrated in figure 8.5. This representation, which shares some similarities with Gantt charts, also

[20] The term breadboard derives from the circuit boards of the same name used for the rapid prototyping of electronic designs.

lends itself to automated generation, unlike the geometric representation as shown in figure 8.2.[21]

8.3.6 Graph support and algorithms

This section outlines the graph support and algorithms used in *deeco*. It is intended to be descriptive rather than comprehensive. The algorithms themselves are marked on the right side of the *deeco* flow chart, given as figure 8.1. The algorithms are not restricted to planar networks.

Graph support

Graph support in *deeco* is via the USL Standard Components 3.0 compile-time libraries.[22] These graph libraries pre-date C++ templates and are therefore called using preprocessor macros.[23] *deeco* makes use of graph algorithms for traversal and searching for cycles.[24]

Exergy interconnection and data-flow digraphs

The complete exergy interconnection digraph, described in section 8.3.3, is enduring.[25] By contrast, the data-flow digraphs, particular to each of the three algorithms, are, effectively, constructed anew at each invocation. But as discussed earlier, these data-flow digraphs never exist as true program entities, instead they simply offer convenient explanations for the operation of the various algorithms as they traverse the exergy interconnections digraph object.

Thermal sub-network (TSN) algorithm

The TSN algorithm is described in detail in section 8.6. This algorithm makes use of a graph class traversal procedure in order to resolve the flow attributes of the thermal sub-networks prior to the TLFO algorithm.

Time-local flow optimisation (TLFO) algorithm

The TLFO algorithm is based on the simplex method. The problem structure is discussed in section 6.5.4 and the simplex method itself described in Foulds (1981).

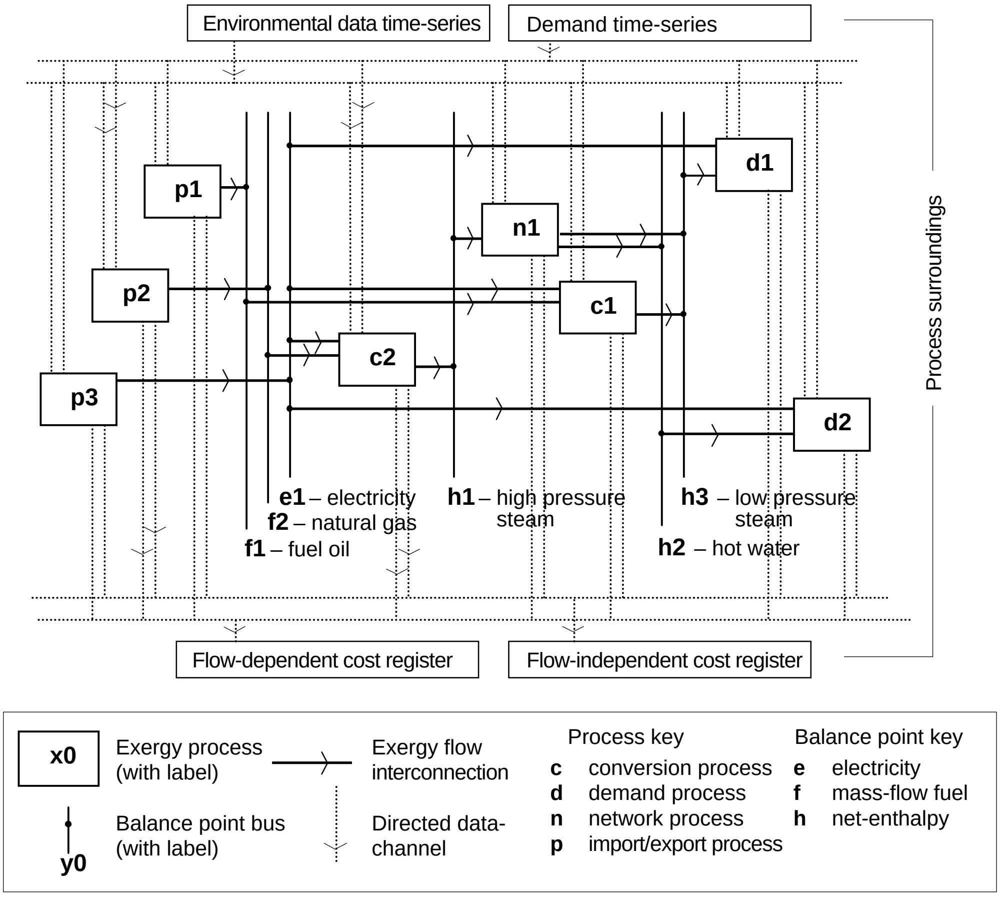
[21] Two graphical user interface (GUI) systems were successfully coded by the writer using MATLAB 5.3 — one was an automatic breadboarding scheme as discussed, and the other, a geometric representation mouse-click-driven network definition procedure.

[22] SCO UnixWare 2.1.x and above bundles the USL Standard Components as release 3.1, due to minor modifications for the port to SDK.

[23] Templates implement the concept of genericity, whereby a range of instance specifications are covered by one class declaration.

[24] A worked example from the SC graph class tutorial is included in appendix C.

[25] A dynamic exergy interconnection digraph would be possible under the *deeco* framework, but would offer no advantage.



Notes:
1. The labelling notation is selected for readability and does not align with the ASCII input data file-set format.
2. The labelling scheme ascends top to bottom for processes and left to right for balance points.
3. The directed data-channels associated with the exergy flows are omitted for clarity.
4. The exergy flow type instances (*eg*, fuel oil) given on the balance point buses are indicative only.
5. The electricity bus **e1** is virtual and does not require explicit definition.
6. The exergy flows shown may be set to zero during the time-local network optimisation algorithm.

Figure 8.5 *A breadboard representation of a network problem. The vertical data-channels would not normally be shown.*

The code itself is from Press *et al* (1993).[26] And, as noted, flow network problems can be tackled using graph algorithms or matrix manipulation. *deeco* uses a graph traversal of the optimisation exergy interconnections digraph to create a suitable problem definition matrix which is then passed to the simplex routine for linear optimisation. An equivalent algorithm can be framed in graph-theoretic terms — in this case, the procedure repeatedly seeks cost augmenting paths until no more can be found.[27] The data-flow digraph which carries the exergy supply requests is identical but anti-parallel to the complete exergy interconnections digraph.

Storage replenishment and export (SRE) algorithm

The SRE algorithm searches for zero-selected-cost storage replenishment and export opportunities, in that order. The algorithm then updates the storage process state information, and export process cost information, as appropriate. As discussed elsewhere, across-interval storage optimisation is not attempted.[28]

8.4 Mathematical representation of processes

This section discusses, in turn, three distinct but interrelated topics: the mathematical representation of processes generally, the evolution of this representation as the model proceeds, and the task of creating new representations for new process types.

The material in this section is similar to that found in discrete-time state-space control theory, although no direct attempt has been made to interpret the ideas presented here in that context. (Luenberger 1979).

In many respects, it is best to approach the mathematical representation of processes in two passes — initially, under the simplification of an invariant process state, and, once understood, adding in the complication of variable process state. Variable process states allow the model to support dynamical systems interactions.[29] This section, in particular, could be read in conjunction with Bruckner (1997, translation). Readers may also wish to consult figure 8.6 (see p 144) in advance.

8.4.1 Process representation

The state and behaviour of real processes needs to be represented mathematically to enable their inclusion in the model domain. In *deeco*, processes are treated as ′black boxes′ as depicted in figure 8.4 — hence their inner operation may be arbitrarily

[26] Numerical Recipes sub-chapters and C source code may be downloaded from the web URL: `http://www.nr.com`.

[27] A listing of the out-of-kilter algorithm trialled by the writer is included in appendix D.

[28] Details of the SRE algorithm are currently unclear to the writer, primarily due to a lack of documentation in English.

[29] Dynamical systems are discussed in footnote 10 (see p 5).

complex and need not be understood. Under this representation the processes themselves become abstractions which respond to data inputs and which provide data outputs. Two core sets of equations are used to describe the behaviour and state of these processes, respectively:

- **input-output equations** — which describe process outputs as functions of process inputs and process state,
- **state-transformation equations** — which describe changes in process state as functions of process inputs and the process state carried over from the preceding time-interval — this latter effect classifies the model as a ′dynamical system′.

The following optional set of equations is provided in addition to those above:

- **threshold equations** — which describe preconditions that need to be met or exceeded before a process can be invoked — the upper capacity is mandatory and is contained within the input-output equations.

These sets of equations, in their most general form for each process type, are encoded into process modules at compile-time. The sets of equations contain parameters which are fixed at run-time using information supplied as part of the process definition. These parameters contain information about the actual process being modelled and may be both interval-independent and interval-dependent. During execution, these sets of equations remain dynamic, in the sense that they are repeatedly re-established in form over the lifetime of a process in response to new information received by that process.

The three sets of equations have the following general appearance, expressed in time-discrete format:

$$\begin{aligned} \mathbf{y}_t &= f(\mathbf{x}_t, \mathbf{z}_t, t) && \text{input-output equations} && (8.1) \\ 0 &\geq f_{\mathrm{T}}(\mathbf{x}_t, \mathbf{z}_t, t) && \text{threshold equations (optional)} && (8.2) \\ \Delta\mathbf{z}_t = \mathbf{z}_t - \mathbf{z}_{t-1} &= g(\mathbf{x}_t, \mathbf{z}_{t-1}, t) && \text{state-transformation equations} && (8.3) \end{aligned}$$

where: $\mathbf{x}$ is the process input variables vector
$\mathbf{y}$ is the process output variables vector
$\mathbf{z}$ is the process state variables vector
t is the interval number, representing discrete time

The process output variables are determined solely by the process, whereas, the process input variables arise from sources external to the process (examples are given in figure 8.6 below the iteration line). Process state variables represent the internal state of the process itself, and are given by eqn (8.3), a first-order difference equation. Process state variables fall into two types: first, variables which are *not* used as output variables, *eg*, process energy demand and useable waste-heat flow, and second, variables which are *also* output variables in their own right, *eg*, storage inventory.

Under *deeco*, the individual equations, which make up a set of equations, perform *different tasks* at the various steps through an iteration. Subscripts are applied to the individual equation function symbols to indicate the task to which they relate — the same subscripts are also used to notate the resulting dependent variable. The actual equations themselves may be structured in any sensible form, so long as all linearity restrictions are met. Further, the equations may be explicit (as shown above) or implicit functions of t. Examples of notated output variables include (dropping the t subscript for clarity):

y_{Att} indicates a flow attribute, output as a scalar
y_{Fl} indicates an exergy flow, output as a scalar

Similarly, examples of input variables include:

$\mathbf{x}_{\mathrm{Att}}$ indicates a set of flow attributes, input as a vector
$\mathbf{x}_{\mathrm{Fl}}$ indicates a set of exergy flows, input as a vector
$\mathbf{x}_{\mathrm{Inf}}$ indicates a set of process external influences, input as a vector

Further discussion is beyond the scope of this thesis, but more detail is available in Bruckner (1997, translation).

8.4.2 Process definition

Programmatically, a process definition creates an instance of a process module by passing parameters supplied as part of the scenario definition. The process is not fully characterised at this point because it lacks the information to do so. But it will be fully specified, although this specification may be interval-specific due to the use of interval-dependent parameters. The process definition parameters are used, for a particular process, to resolve the following:

- its various **energetic performance indexes**, dependent on external influences and process state, but independent of exergy flows,
- its **capacities**, both lower (*ie*, threshold) (the default is zero) and upper, dependent on external influences, but independent of exergy flows,
- process **load-independent exergy demand** (*eg*, to power pumps) and/or supply, by type, dependent on external influences and process state, but independent of exergy flows,
- process **flow-dependent costs**, dependent on external influences and linear on exergy flows,

- process **flow-independent costs**, dependent on external influences and independent of exergy flows — but based on maximum invoked duty,
- process **initial conditions**, defined by the process extensive state and state attributes (*eg*, for storage inventory).

Process energetic performance indexes capture the relationship between entering exergy flows, exiting exergy flows, and the internal extensive state. Processes with co-product or co-fuel streams or storage, will require more than one mathematical statement, as discussed in section 8.4.4.

Process parameters used in a given scenario can be tested for sensitivity by altering one value or time-series and re-running the model.

8.4.3 Process state and behaviour within the model

Process instances are persistent, but the state and behaviour of the individual process are regularly updated, both within and between iterations. The term ′process characterisation′ (introduced on p 131) describes the set of events which results in the state and behaviour of all processes being fully resolved prior to the execution of the TLFO algorithm. These steps are as follows:

- process states are updated in light of the newly identified optimal flow routing regime,
- the SRE algorithm is run, and the process states are again updated as appropriate,
- the interval index t is incremented,
- the input-output, threshold, and state-transformation equation sets are recast, using interval-specific information,
- the TSN algorithm is run to resolve all *flo* and *return* attributes, and to thereby finalise the input-output, threshold, and state-transformation equation sets.

The information interchange within *deeco* is complicated. The *deeco* flow chart, given in figure 8.1 (see p 121) indicates the various interchange events in terms of the program procedure. Figure 8.4 (see p 126) shows these interchanges in terms of the arrangement of directed data-channels relative to each exergy flow type. Figure 8.5 (see p 137) shows these interchanges relative to the process surroundings. And figure 8.6 (see p 144) shows these same interchanges from the perspective of an individual process over time.

deeco uses a first-order Euler method to solve the state-transformation equations numerically, which implies the states are assumed constant during the TLFO algorithm. The states are then updated upon completion of this algorithm, and can be used to influence the underlying marginal performance indexes in a nonlinear fashion.[30]

8.4.4 Process module specification

Process module specification describes the task of taking some exergy process type of interest (*eg*, a particular form of heat-pump) and representing it mathematically in terms of its salient external dependencies. This description, as discussed in section 8.4, together with certain programmatic information such as its process category, provides the means to code a process module. Process modules, once compiled and included in the *deeco* executable, are accessible to subsequent users via the scenario definition.

The input-output, threshold, and state-transformation equation sets are built around what are termed ′marginal ϵnergetic performance indexes′ — with the qualifier ′marginal′, as opposed to ′average′, indicating the partial differential form. In general, these indexes result from the arithmetic ratio of two extensive quantities, and may be formed using process state variables as well as exergy flows. The index most commonly encountered is the marginal ϵnergetic efficiency, which applies to ′fuel in – product out′ processes. This and other examples are shown below in eqn (8.4) (the notation is similar to that used on Bruckner (1997, translation), eqn 2.9, but with the scripting replaced by words (as defined on p 170); the symbols $\lambda, \epsilon, \mu, \sigma$ are specific to Bruckner):

$$\lambda = \frac{\partial \dot{L}_{\text{fuel}}}{\partial \dot{L}_{\text{product}}} \quad \epsilon = \frac{\partial \dot{L}_{\text{product}}}{\partial \dot{L}_{\text{fuel}}} \quad \mu = \frac{\partial \dot{L}_{\text{co-product A}}}{\partial \dot{L}_{\text{co-product B}}} \quad \sigma = \frac{\partial \dot{L}_{\text{co-fuel A}}}{\partial \dot{L}_{\text{co-fuel B}}} \tag{8.4}$$

The above equations should be compared in form with eqn (5.16) (see p 65).

These marginal ϵnergetic performance indexes are derived from stable process performance information, generally obtained from two sources. First, from engineering design formula using empirical constants (*eg*, the log mean temperature difference (LMTD) equation for heat-exchangers). Or second, from measured steady-state characteristic curves published in engineering data-books or by equipment manufacturers. Other information, such as load-independent exergy requirements, also needs to be identified as part of the process type specification exercise.

In summary, the specification of a new module requires mathematical statements of the following (as appropriate):

- the underlying process category (refer to table 8.2),
- the net-enthalpy flow types, if heat-exchange is used (refer to Bruckner (1997, translation), fig 2.3),

[30] Refer to Bruckner (1997, translation), eqn 2.72.

- the dependency of relevant marginal ϵnergetic performance indexes on process input variables (including flow attributes) and process state variables,
- the dependency of all load-independent exergy flows on process input variables (including flow attributes) and process state variables,
- the flow attributes used as process output variables,
- the input-output equations,
- the threshold equations,
- the state-transformation equations.

All mathematical statements will need to comply with the modelling restrictions required by *deeco* as outlined in section 8.4.2. The linearity restrictions are imposed by the TLFO algorithm. Readers seeking more details should consult the *deeco* documentation and/or refer to the source code.

8.4.5 Process lifecycle

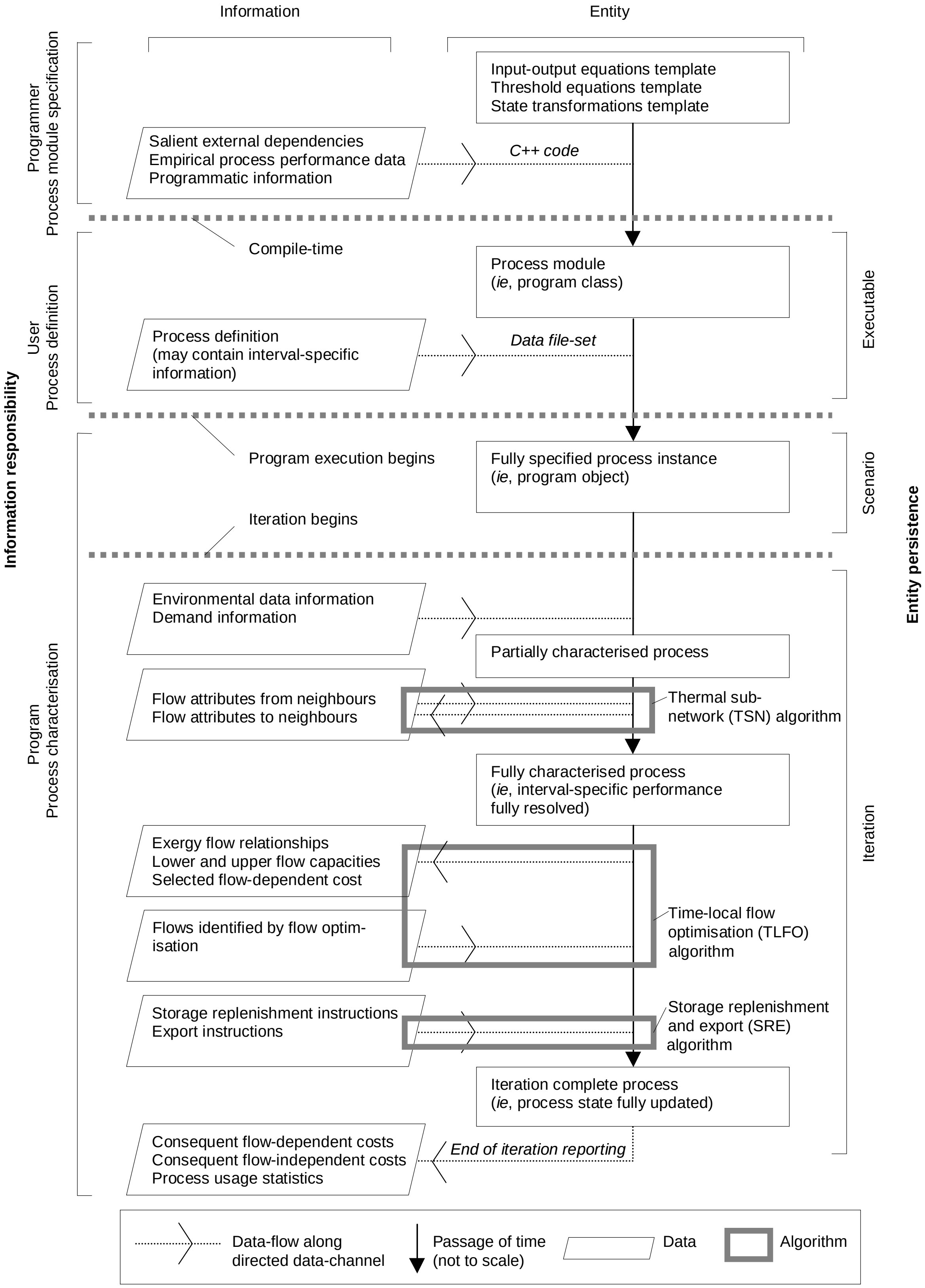
A number of steps are required to abstract a real process and deploy it in the model domain. These steps are summarised in figure 8.6 from the point of view of a process over time. To recap, one interpretation of this sequence of events, setting aside the complication of non-steady-state behaviour (*ie*, storage processes) and heat-exchange (*ie*, flow attribute resolution) is as follows:

- identify the process category, locate typical ϵnergy efficiency relationships from the engineering literature, and generalise this information in terms of design parameters and salient external influences,
- subsequently and within *deeco*, create a desired process by defining these design parameters,
- for each time-interval and the given prevailing external conditions, particularise the ϵnergy efficiency relationships,
- use this information, plus other process details, to inform the TLFO algorithm of the relationship between entering flows, exiting flows, flow capacities, and the selected consequent flow-dependent costs.

8.5 Nonlinear costs and flow-dependent performance

This section outlines an extension to the current version of *deeco*.[31] Section 6.5.6 (see p 96) provides a theoretical context to the following discussion.

[31] This extention was contributed by the writer.


Information
Entity
Programmer
Process module specification
Input-output equations template
Threshold equations template
State transformations template
Salient external dependencies
Empirical process performance data
Programmatic information
C++ code
Compile-time
User
Process definition
Process module
(ie, program class)
Executable
Process definition
(may contain interval-specific information)
Data file-set
Information responsibility
Program execution begins
Fully specified process instance
(ie, program object)
Scenario
Entity persistence
Iteration begins
Environmental data information
Demand information
Program
Process characterisation
Partially characterised process
Flow attributes from neighbours
Flow attributes to neighbours
Thermal sub-network (TSN) algorithm
Fully characterised process
(ie, interval-specific performance fully resolved)
Iteration
Exergy flow relationships
Lower and upper flow capacities
Selected flow-dependent cost
Time-local flow optimisation (TLFO) algorithm
Flows identified by flow optimisation
Storage replenishment instructions
Export instructions
Storage replenishment and export (SRE) algorithm
Iteration complete process
(ie, process state fully updated)
Consequent flow-dependent costs
Consequent flow-independent costs
Process usage statistics
End of iteration reporting
Data-flow along directed data-channel
Passage of time (not to scale)
Data
Algorithm

Notes: 1. Information internal to the process is not shown. 2. The interation loop is not shown to assist clarity.

Figure 8.6 *The lifecycle of a process.*

Nonlinear costs

Convex and strictly increasing nonlinear costs can be used in *deeco*.[32] Such cost functions need to be approximated in piecewise linear form, as shown in figure 8.7. Concave costs, by contrast, cannot be used because these can frustrate the search for a true minimum.

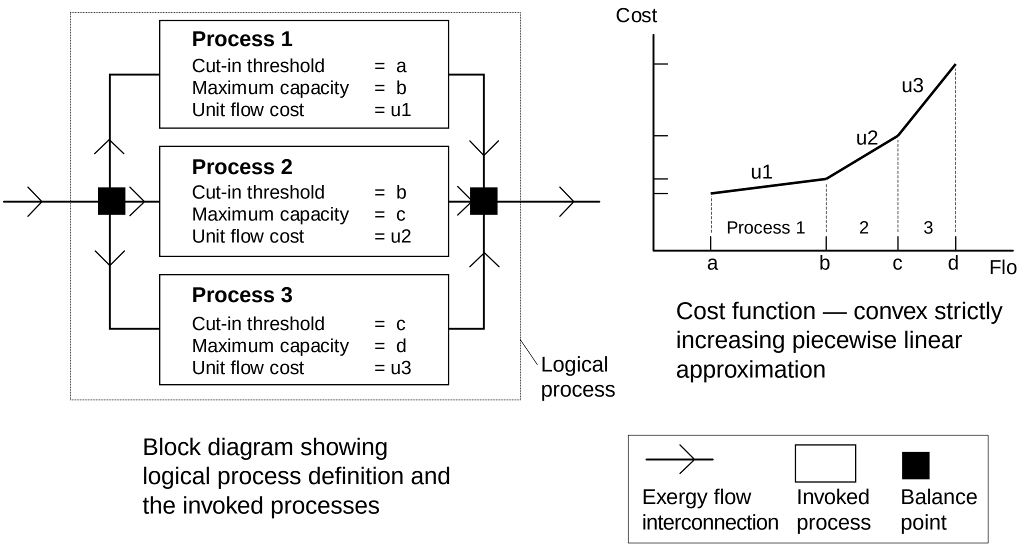


Figure 8.7 *Piecewise linear cost function used to approximate a convex strictly increasing nonlinear cost function.*

Flow-dependent performance

A similar strategy can be used to provide flow-dependent process performance. The technique is similar to that shown in figure 8.7 except that the invoked process sequence is modified so that process efficiency, rather than (the inverse of) cost, is recast downward at each successive kink. This method allows for the modelling of transmission line losses, as described on p 165.

Implementation

These changes can be implemented without modifying the executable by including them within the scenario definition. But it may be possible to implement this facility as a logical process module (*ie*, class) that, when instantiated, spawns a defined

[32] A function, over a given interval, is ′convex′ *iff* a line connecting two points sits fully *above* the curve. A function, over a given interval, is ′strictly increasing′ *iff* for every $x_1 < x_2$ gives $f(x_1) < f(x_2)$.

number of invoked processes (*ie*, objects) in parallel and with the characteristics required. Note that the invoked processes are still linear in themselves.

8.6 Thermal sub-networks

The TSN algorithm completes the characterisation of the network in readiness for the TLFO algorithm.

8.6.1 Thermal sub-network definition

Any group of processes linked by heat flow are deemed to constitute a ′thermal sub-network′. The principal method of heat exchange between plant in most exergy systems is convective heat transfer, forced by means of electrically-driven feed pumps. Actual heat transfer, which comprises conduction and radiation, is treated programmatically as a special case of convective transfer.[33] Convective heat transfer requires *flo* and *return* piping — with the *flo* connection being that which carries the transfer medium parallel to the exergy flow.

8.6.2 Net-enthalpy flow

Net-enthalpy flow (**H**) is used to describe heat-exchange and is accounted for in *deeco* using flow ϵnergy (as is all exergy flow). Net-enthalpy flow (expressed in time-continuous form) is given by:[34]

$$\dot{L}_{\mathrm{H}} = \dot{m}(h_{\mathrm{F}} - h_{\mathrm{R}}) \tag{8.5}$$

where: $\dot{L}_{\mathrm{H}}$ is the net-enthalpy flow-rate
$\dot{m}$ is the mass flow-rate of the circulating transfer medium
h_{F} is the mass-specific enthalpy of the *flo* flow
h_{R} is the mass-specific enthalpy of the *return* flow

Equation (8.5) is used to contract the two heat transfer streams into a single exergy flow. In this regard, net-enthalpy flow is a fiction, but one necessary to create a single- rather than multiple-edged complete exergy interconnection digraph. For this reason, net-enthalpy exergy flow is also described as logical exergy flow (see p 71). Under certain circumstances, processes linked by net-enthalpy flow are able to influence the characterisations of their neighbours — this is described shortly. Further, a process which has the flow attributes of all connected exergy streams resolved, is necessarily fully characterised.

[33] See Bruckner (1997, translation), definition 7.

[34] This expression can be stated in more general form using ϵnergy: $\dot{L} = \dot{m}(\ell_{\mathrm{F}} - \ell_{\mathrm{R}})$

The steps required to represent convective heat transfer between two neighbouring processes as a single net-enthalpy flow are depicted in figure 8.8. It is necessary for the transfer medium circulation time to be significantly less than the time-interval in use (the default being 1 hour).

8.6.3 Flow attribute setting

The flow attributes are the intensive values of the appropriate *flo* and *return* streams, and may include, for example: temperature, pressure, and dryness fraction (in the case of partially-condensed steam).

The TSN algorithm is based on the understanding that certain process types can dictate to other process types, some or all of the flow attributes that they require, and/or are willing to accept. This, therefore, involves communication between processes. The communication channel may comprise the actual *flo* stream or *return* stream, or alternatively, it may require, or be, a dedicated circuit, typically using analogue or digital electronics.

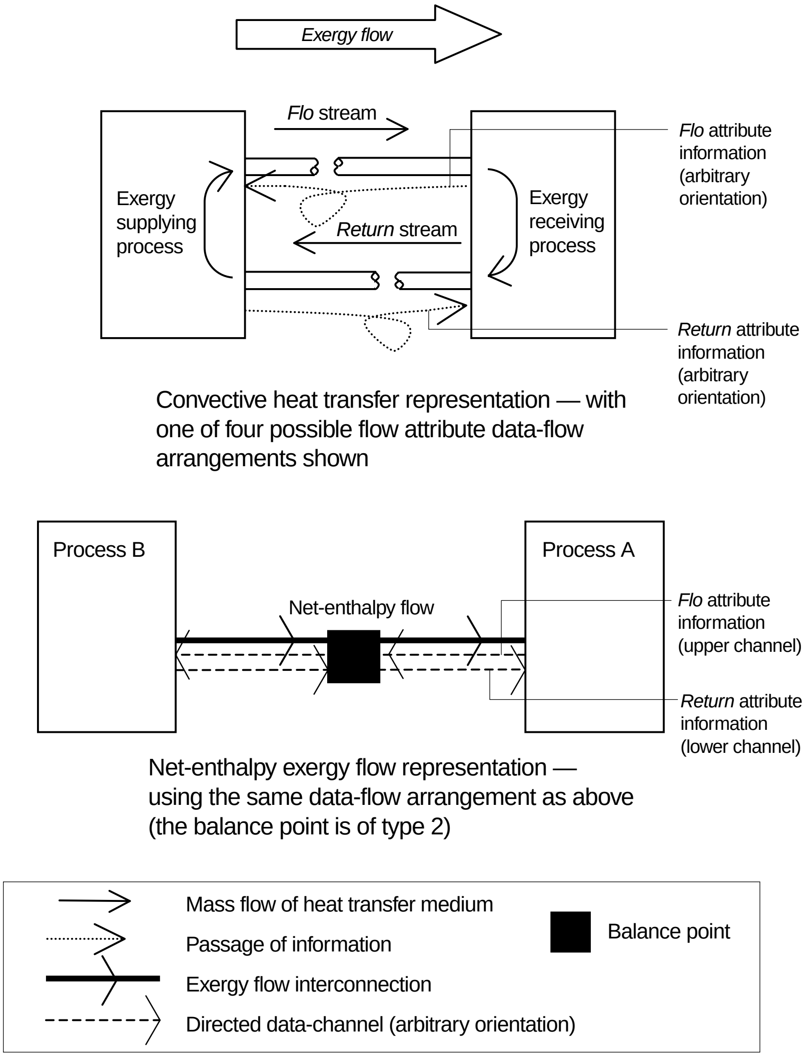
The plant being modelled needs to possess sufficient active control to enable the TSN method — in other words, the algorithm attempts to capture the communication and control protocols found in such circumstances. Not all heat-exchanging plant can be represented thus, but most of the networked equipment found in exergy systems will comply.

Much of the complexity of the algorithm is not contained within the actual procedure, but rather is concerned with restricting the behaviour of process modules and the arrangement of process interconnections so that the method can be applied. The algorithm itself is based on a two-pass graph traversal, is non-iterative, and gives a unique result.

The key requirement is that processes can set and transmit some or all of the *flo* and *return* attributes unilaterally *and/or* receive and use some or all of the *flo* and *return* attributes compliantly, but *only in a manner* whereby circular dependencies are avoided. Note that some process types can co-produce net-enthalpy flow. Bruckner (1997, translation), fig 2.10, shows exactly which internal dependencies between received and sent *flo* and *return* attributes are permitted and which are illegal.

Each *flo* or *return* attribute sent is a process output variable and must be uniquely derived from the process input-output equations set (see eqn (8.1)) using the process input variables, which may include received *flo* and/or *return* attributes, as well as process state variables.

Further limitations apply where the net-enthalpy flows split or join, as shown in figure 8.9. The purpose of these restrictions, collectively, is to avoid the following


Exergy flow
Flo stream
Exergy supplying process
Return stream
Exergy receiving process
Flo attribute information (arbitrary orientation)
Return attribute information (arbitrary orientation)
Process B
Process A
Net-enthalpy flow
Flo attribute information (upper channel)
Return attribute information (lower channel)
Mass flow of heat transfer medium
Passage of information
Exergy flow interconnection
Directed data-channel (arbitrary orientation)
Balance point

Convective heat transfer representation — with one of four possible flow attribute data-flow arrangements shown

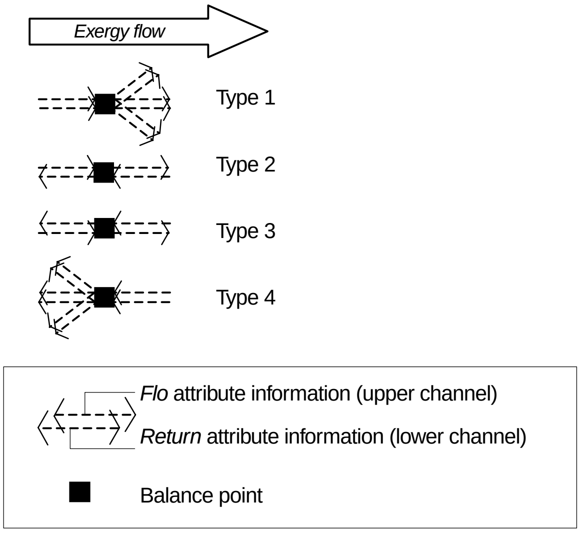
Net-enthalpy exergy flow representation — using the same data-flow arrangement as above (the balance point is of type 2)

Notes:
1. Net-enthalpy flow is defined as being parallel to exergy flow, not energy flow.
2. The passage of information can be via the transfer medium if circumstances allow, otherwise a dedicated channel is required.
3. It is process A which demands the net-enthalpy flow from process B during the flow optimisation step.

Figure 8.8 *Convective heat transfer recast as a single net-enthalpy (or logical heat) flow.*

conditions:

- two or more processes issuing conflicting directives to another process,
- unresolvable circular dependencies involving two or more processes,
- any process failing to receive sufficient information to complete its characterisation.


Exergy flow
Type 1
Type 2
Type 3
Type 4
Flo attribute information (upper channel)
Return attribute information (lower channel)
Balance point

Notes: 1. The use of data-channels as shown is specific to the TSN algorithm.

Figure 8.9 *Allowable thermal sub-network balance point types.*[35]

In practice, users need only concern themselves with the applicability of the process modules selected and ensuring that the connectivity restrictions are complied with.

8.6.4 Thermal sub-network (TSN) algorithm

The TSN algorithm operates on its own data-flow multi-digraph, which overlays the restricted EID, and also on the restricted EID itself.[36] In this case, two directed data-channels are mandatory, for *flo* and *return* attribute information, respectively, and parallel or anti-parallel to the exergy interconnection as dictated by the scenario definition. The restricted EID is necessarily acyclic, due to modelling restrictions discussed on p 124. The algorithm traverses the restricted EID twice, resolving process characterisations partially or fully by passing *flo* and/or *return* attribute information between adjoining processes as it progresses. The traversal starts from the set of processes which ′annihilate′ exergy flows, *ie*, demand and export processes

[35] Based on Bruckner (1997, translation), fig 2.11.

[36] Restricted exergy interconnection digraphs are discussed on p 124.

(see p 127), advances upstream, and, when finished, restarts from the termination vertices and works downstream in reverse order. As a side-effect, the algorithm identifies two multi-root-vertex spanning out-trees (see figure 8.10), one parallel and the other anti-parallel, on the restricted exergy interconnections digraph.[37] Further, these two out-trees, when merged (*ie*, unioned) form a (not necessarily connected) subgraph which captures only the thermal sub-networks, and contains only balance points of the type shown in figure 8.9. The double traversal procedure, and the detailed actions undertaken at each vertex are described fully in Bruckner (1997, translation), section 2.5. Once the thermal sub-networks have been resolved, the virtual data-flow multi-digraph is abandoned.

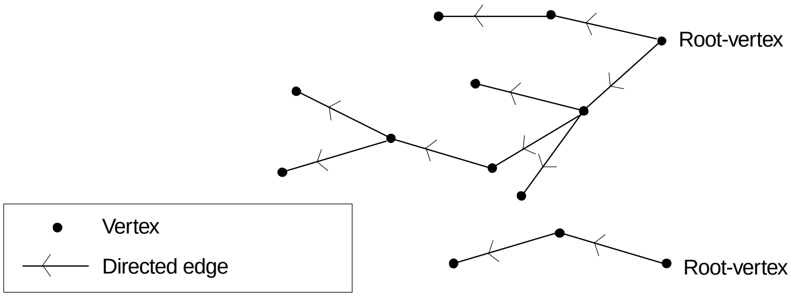


Figure 8.10 *A generalised multi-root-vertex spanning out-tree. As a result of the TSN algorithm, deeco produces two such trees as parallel and anti-parallel subgraphs of the restricted exergy interconnections digraph.*[38]

8.6.5 Closure

The TSN algorithm does not optimise for *flo* and *return* attributes, or for process performance. Rather, the algorithm allows certain collections of processes to set the *flo* and *return* attributes in a manner which is tractable under a two-pass procedure, and which can capture the control behaviour found in real situations. It is a central premise of *deeco* that these flow attributes, once set, remain unaltered by the exergy flows determined during flow optimisation, or subsequent storage replenishment and export.

[37] The spanning out-tree structure is identical to that used in Leontief input/output economic modelling (see p 83).

[38] The multiple root-vertices can be fed from a single virtual root-vertex, so that a multiple-root-vertex spanning out-tree can always be re-represented as a single-root-vertex spanning out-tree. An out-tree is also known as an arborescence.

8.7 Quality of model issues

In this section, the term ′model′ includes both the application program *deeco* and some given *deeco* scenario.

Model validation can be seen as two-stage. First, program ′verification′ establishes that the numerical program captures and executes the model abstraction as identified. Second, model ′validation′ additionally requires that the model abstraction is sufficiently representative to fulfil the purposes of the model — in this sense, validity is context-dependent.

8.7.1 Program verification

In practice, program verification can only be framed in the negative, meaning that, given a certain amount of analysis and testing, no problems came to light. Robust verification is a substantial topic, so the following treatment is necessarily incomplete.

The starting point for program verification is a well articulated program specification. In practical terms, Bruckner (1997) together with the available user and programmer documentation can be interpreted as that specification.

The core algorithms used in *deeco* have mathematical proofs showing that they complete in all circumstances, provide unique and repeatable results from a given data-set structure, and are free from convergence and stability issues. It follows, therefore, that spurious output or failure to terminate are not algorithm design issues for *deeco*. *deeco* contains a number of checks on data integrity, and will abort if problems (such as cyclic structures) are detected. The source code for *deeco* is well constructed, but would be very time-consuming to review line-by-line.

8.7.2 Modelling assumptions and restrictions within *deeco*

deeco contains a number of assumptions and restrictions that need to be confirmed prior to use. The principal issues are summarised below:

- processes exchanging heat require certain behaviours and restrictions in terms of neighbour interaction, internal behaviour, and allowable connectivity,
- process marginal ϵnergetic performance indexes are independent of the exergy flows,
- flow attributes are independent of the exergy flows, but may be dependent on the extensive state of the connected processes (*eg*, storage inventory),
- the selected flow-dependent cost must be linear on exergy flows,

- notwithstanding the above, approximations can be used for convex, strictly increasing nonlinear costs and/or process performance,
- cyclic structures, except within the electricity pool, are prohibited,
- import/export transactions do not affect the flow-dependent costs associated with those transactions.

8.7.3 Model validation

Model validity concerns itself with the degree of fit between model abstraction and reality. Such assessments can only be conducted on a case-by-case basis, and in terms of the purpose of the model.

However, taking a more general view, *deeco* is widely applicable. Energy industry plant often exhibits efficiencies which are near-independent of load, particularly those that run parallel units (*eg*, Huntly power station operates 4 × 250 $\mathrm{MW_e}$ units). Heat-exchanging plant is often controlled in a manner which allows the TSN algorithm to apply. And quadratic electricity transmission losses can be captured using nonlinear performance.

Finally, it should be borne in mind that *deeco* is primarily a policy assessment tool, not an engineering design tool, and thus is not intended to substitute for detailed engineering and/or financial calculations.

8.8 Programming issues

8.8.1 Object-oriented programming

The object-oriented programming (OOP) methodology is ideally suited to *deeco*. Process modules naturally represent as classes and their similarities can be leveraged through inheritance. Further, the network structure can readily be implemented using graph-based container classes, available as compile-time libraries which also bundle useful functionality (*eg*, traversal) and, in some cases, optimisation routines (*eg*, MCFP).

C++ is an obvious choice for the programming language, both at the time of original development and currently. The principle drawback for C++, aside from its difficult syntax, is the inconsistency of language implementation — most code of any complexity is either compiler-specific or carries the development overhead of conditional compilation.

8.8.2 Software development environment

deeco was originally developed on HP-UX 8.07 / RISC. It was ported to SCO UnixWare 2.1.2 SDK / x86 as part of this thesis.[39] These two development environments are compared in table 8.6, with abbreviations given on p xi. Note that SCO UnixWare has a very different lineage from the more common SVR3-based SCO UNIX, although both operating systems were marketed concurrently by SCO for approximately 3 years.

deeco is written in a *circa* 1990 C++ dialect known as 'ARM' because it aligns with that described in the Annotated C++ Reference Manual (Ellis and Stroustrup 1992). *deeco* was originally built using what is essentially the AT&T `cfront` 3.0.1 C++ front end. The Edison Design Group (EDG) C++ front end bundled with UnixWare, to which *deeco* was ported, is well regarded. Details can be found in Edison Design Group (1998) as well as in the UnixWare operating system documentation. The SDK release of EDG compiler can be back-loaded onto UnixWare 7.0 UDK if required.

As table 8.6 shows, the port to UnixWare was clearly indicated. The new compiler contains compatibility support for the original compiler dialect, the component libraries are functionally identical, and the application programming interface (API) is similar. However, version 0.5 of *deeco* did contain environment-dependent code concerning floating point exception handling and this code needed revising.

The following references were particularly useful when porting *deeco*: Curry (1996), Gilly (1992), Goodheart and Cox (1994), Lehey (1995), Lewine (1991), Stevens (1992), Stroustrup (1991), UNIX Systems Laboratories (1992), Weitzen (1992).

8.8.3 Application details

Copyright

Copyright is held by Dr. Thomas Bruckner.

Fitness for purpose

Dr. Bruckner advises that *deeco* is without warranty of any kind including that the program is free of error.

Software metrics

deeco consists of 18 000 lines of code, including comments, and the listing runs to 295 pages. *deeco* is fully object-oriented and makes use of 66 user-defined classes with 3-deep inheritance. The component libraries, API calls, user-defined classes

[39] The term 'x86' indicates an Intel-compatible 386DX or higher processor.

Attribute		**HP-UX 8.07 / RISC**	**UnixWare 2.1.2 SDK / x86**
Operating system details	OS vendor:	Hewlett-Packard	SCO (Santa Crux Operation)
	UNIX name:	HP-UX	UnixWare
	Version used:	**8.07**	**2.1.2**
	UNIX base:	SVR3.2 with BSD extensions ?	SVR4.2 MP
	Major version release:	HP-UX 8.0	Novell UnixWare 2.0 (later sold to SCO)
	Shipping date:	circa 1991	December 1994
	Default shell:		KornShell (ksh)
	OS release history:	8.0 1991	2.0 Dec 1994 (by Novell)
		8.07 1991	2.1 > Dec 1995 (by SCO)
			2.1.2 April 1997 (no OS changes)
	OS features of interest:		dynamically loadable kernel modules multi-processor support Veritas journaling file system (vxfs)
	Contemporary OS:	HP-UX 10.x	UnixWare 7.0.x
	Described OS withdrawn:	circa 1995	July 1998
	Contemporary OS notes:	new HP aC++ compiler will *not* support old code	UnixWare 7 UDK is SVR5-based legacy code *should* generally compile
Hardware	Hardware vendor:	Hewlett-Packard	AMD
	Hardware specs:	RISC 700 series workstation	200MHz K6 chipset (Pentium Pro equivalent) 32 MB RAM
Development environment	Development environment:	HP-UX	SDK (Software Development Kit)
	Bundled compiler:	USL C++ translator	EDG C++ front-end
	Version:	**3.0.1**	**2.1.2** (versioned as per OS release)
	Language base:	ARM	ARM plus extensions
	Command line:	CC	CC
	Notes:	essentially AT&T cfront 3.0.1	support for 3.0.x cfront dialect
	Contemporary support:	no, discontinued, now aC++	yes, compatibility support under UDK
	Graph libraries:	USL C++ Standard Components	SCO C++ Standard Components
	Form:	HP binaries and source	SDK binaries only
	Version:	**3.0**	**3.1** (minor modifications for SDK port)
Published standards	POSIX 1003.1 (systems calls)	unlikely	conformance claimed from 2.0.x
	IEEE 745 (floating point arithmetic)	yes — but defaults and implementation may vary	yes — but defaults and implementation may vary
	SPEC 1170 (API standard)	not known	certification sought for 2.1.1

Table 8.6 *The original and ported development environments.*

and dependencies, and internal calls cross-referencing, were identified as part of the original code evaluation, but space limitations prevent their inclusion here.

Execution times

For hourly/annual municipal problems, execution times are in the order of 60 min, so each iteration completes in about 400 ms. Preliminary trials indicate the 200 MHz Intel-equivalent platform is at least 70% faster than the original HP 700 series workstation. Memory availability (*ie*, algorithmic space complexity) does not appear to be an issue.

Application documentation

At the time of writing, user and developer documentation and related support material was incomplete and mostly in German:

> User's manual. Part I – How to prepare input files. 19 December 1997. (18 pages, German). Part II – Technical description of processes. 02 April 1996. (139 pages, German).
>
> Programmer's manual. Part I – Libraries used. 19 December 1997. (9 pages, English). Part II – Flow charts, variable names. 29 March 1996. (51 pages, German).

The SOLEG project (see p 161), funded by the Bavarian Research Foundation, Germany, is currently expanding the documentation, but this will remain in German.

8.9 Future development options

As it stands, *deeco* exhibits several addressable shortcomings, namely:

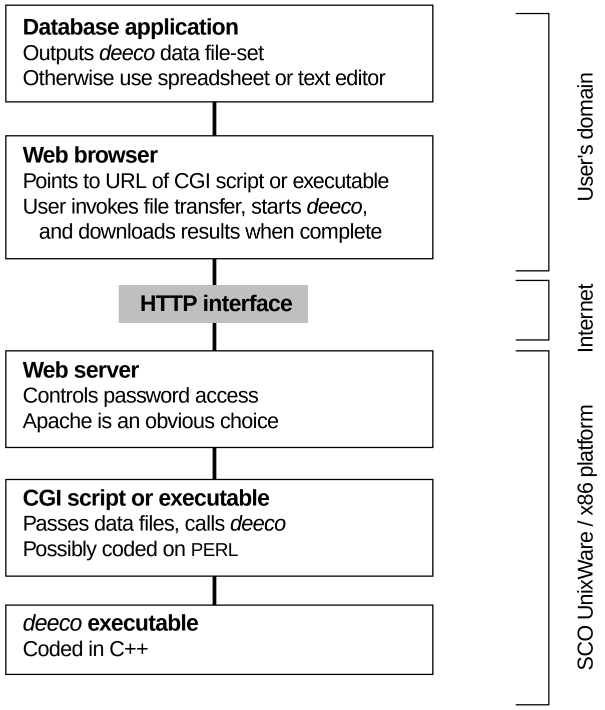
- use of a specialist platform — SCO UnixWare 2.1 or better,[40]
- high-overhead data management — particularly concerning the preparation and interpretation of input and output files,
- the application is primarily structured to deal with municipal problems,
- incomplete user and developer documentation.

Matters arising from the first two points are discussed next.

[40] UnixWare Intel binary compatibility standards suggest that SDK executables should run on other Intel platforms. A version 0.6 *deeco* binary was executed on a Linux machine, but crashed immediately.

8.9.1 Web-based deployment with user-domain data management

A proposal for web-based deployment, using an HTTP-based client/server architecture, is shown in figure 8.11.


Database application
Outputs *deeco* data file-set
Otherwise use spreadsheet or text editor
Web browser
Points to URL of CGI script or executable
User invokes file transfer, starts *deeco*,
and downloads results when complete
HTTP interface
Web server
Controls password access
Apache is an obvious choice
CGI script or executable
Passes data files, calls *deeco*
Possibly coded on PERL
deeco executable
Coded in C++
User's domain
Internet
SCO UnixWare / x86 platform

Notes: 1. CGI stands for common gateway interface.
2. Apache is the name of an OSS web server application.

Figure 8.11 *Proposed deployment architecture for deeco.*

This proposal embraces the ′thin-client′ philosophy and its attendant characteristics. For the user, web-based deployment means *deeco* can be run without installing a specialist operating system. And for the developer, updated executables and important information can be disseminated with minimal overhead. In addition, the scheme allows for a private newsgroup (*eg*, by discussion server) to foster dialogue amongst users and developers. Furthermore, users retain responsibility for their own data management — this provides users with the option of writing custom database applications to prepare and interpret *deeco* data file-sets in accordance with their specific needs.

8.9.2 Software development model

deeco, like any software project, requires ongoing use and development to keep it relevant and viable. The open-source software (OSS) development model, as described by Raymond (1998a,b), probably offers the most promising development strategy. However *deeco* is clearly more specialist than other OSS successes, such as Linux and PERL. Even so, a small development community has already arisen around *deeco* and this should, with luck, continue to grow.

8.9.3 Major redevelopment options

Several 'blue-sky' development options present themselves and are described below. Potential methodological extensions, as discussed separately in section 6.5.6 (see p 96), are not repeated here.

Multiple environmental zones

Multiple environmental zones could be included to account for geographic distance effects within a given scenario. Each zone could contain its own local electricity pool, with inter-zone electricity transmission modelled using quadratic losses.

Second law process performance reporting

deeco could identify exergy quality factor discrepancies across processes and calculate process-specific exergy destruction rates.[41] Assuming monetary information is available, *deeco* could calculate process-specific thermoeconomic indexes.[42] *deeco* could also search for and highlight the worst offending processes — information which could help guide subsequent redesign efforts.

Graphical representation

A graphical infrastructure representation using the 'breadboard' scheme shown in figure 8.5 could be used to visualise the scenario data before submission and then overlay the scenario results upon completion. The feature would probably be best implemented using the *deeco* ASCII data file-sets as the interface protocol. If web-based deployment were adopted (see p 156) then the graphics chould be generated as dynamic HTML.

Smart storage

It may be useful to incorporate smart storage into *deeco*, probably using techniques from dynamic programming (DP) as discussed in section 6.6.1. However, a similar intertemporal optimisation strategy would need to be present in the actual network under investigation in order to capitalise on any duly identified benefits.

[41] The flow exergy quality factor Z is given in eqn (5.17).

[42] Thermoeconomic indexes are discussed in page 49.

Rewrite in ISO standard C++

There is merit in rewriting *deeco* in ISO standard C++ [43] using a contemporary compiler such as GNU g++ / Linux. Unfortunately, the current ISO standard libraries do not include graph support, so a third party source would be required. Two promising options are LEDA and NetClass, both described in appendix A. LEDA has the advantage of containing algorithms for MCFP, with the relevant member function declaration being (Mehlhorn and Näher 1999):

```
bool MIN_COST_FLOW(graph& G, const edge_array<int>& lcap,
                             const edge_array<int>& ucap,
                             const edge_array<int>& cost,
                             const edge_array<int>& supply,
                                   edge_array<int>& flow).
```

8.10 Recap

To recap, *deeco* is constructed as follows. An exergy-services supply system is represented as a flow network problem using a graph structure to contain the processes and their interconnections. Each process determines its own efficiencies and supply potential in light of the prevailing conditions, and the network as a whole faces time- and location-specific demand for services. Following selection of a cost goal, the program iterates through a sequence of interval-averaged time segments, and, at each iteration:

- resolves flow attributes (*eg*, exchange media temperatures) for all thermal sub-networks,
- determines a least-cost flow routing, using storage and surplus co-production (*eg*, waste-heat) as appropriate,
- replenishes storage and exports, in that order, given that no selected costs accrue.

When the time-horizon is complete, the program halts and reports decision-useful information to the modeller.

The key conceptualisations within *deeco* which allow real exergy-services supply systems to be abstracted in a numerically tractable, representative, and useful manner in order that sustainability opportunities can be explored, are as follows:

- the framing of exergy-services supply system integration as a flow network optimisation problem (time-local flow optimisation (TLFO) algorithm),
- the use of sustainability costs as optimisation goals in their own right, rather than as constraints in a financial problem (ethical behaviour),

[43] ISO/IEC 14882 — Standard for the C++ Programming Language.

- the interpretation of demand in terms of exergy-service rather than exergy *per se* (demand process specification),
- the method of generalising and then particularising process performance within the model in light of the prevailing environmental and network conditions (interval-specific process characterisation) and prior state (dynamical systems behaviour),
- the method of allowing communicating heat-exchanging processes to determine their collective state in a manner which mimics the control behaviour of commonly-encountered systems (thermal sub-network (TSN) algorithm),
- the method of storage replenishment and export (storage replenishment and export (SRE) algorithm).

8.11 Comparison with similar models

A number of 'energy' systems optimisation models embed a network representation, although the writer was only able to locate one with more than a tenuous similarity to *deeco* in terms of structure and/or application — this model is MODEST, and is described elsewhere (see p 46).

More generally, *deeco* contrasts with the optimisation-based models reviewed in chapter 4 in that it supports high spatial and temporal resolution. *deeco* is therefore able to capture important correlations within the demand profiles in context of the network infrastructure and as influenced by environmental conditions. All other models, including MODEST, rely on statistical techniques and process aggregation to 'collapse' the problem, and arguably lose important fine-grain informational structure as a consequence.

8.12 Emissions credit evaluation

Dr. Helmuth Groscurth is currently reviewing the use of *deeco* for evaluating CO_2-e emissions credits.[44] In the event that tradable emission permits are introduced, firms could well be entitled to emissions credits if they can demonstrate that their post-1990 plant upgrades and/or operations management improvements exceed their BAU (business-as-usual) net CO_2-e emissions trajectories. There is some debate at present as to whether the BAU case is best defined in terms of actual practice or best-practice — the latter approach prevents poor performers being 'rewarded' by being grandparented additional entitlements, but retrospective equity favours the

[44] Personal comment by email, 14 January 2000 and 02 February 2000. Dr. Groscurth is also participating in EUROTECTRIC's greenhouse gas trading simulation pilot. EUROTECTRIC is the association of European electricity companies.

former approach. *deeco* is well suited to help determine the best-practice BAU benchmarking and to quantify improvements relative to this point. *deeco* is also well placed to assist with post-allocation CO_2-e management.

8.13 Published studies using *deeco*

Two large-scale studies have been published using *deeco*, both tackling problems at the municipal level.

8.13.1 Würzburg study

The Würzburg study analyses municipal energy system investment options for the south German city of Würzburg — with a population 130 000 and using 1993 demand and weather data. The study is not sufficiently representative to allow direct use of the conclusions. But the study clearly demonstrates synergy and counteraction between different technologies given the particular demand profile — shown in location-aggregated form in figure 8.12. Of note, medium-scale co-generation district heating and domestic insulation reinforced each other, yielding a 48% saving in depletable exergy use with a monetary cost penalty of 15% over the *status quo*. Some results from this work are depicted in figure 1.3 (see p 18). (Bruckner *et al* 1997).

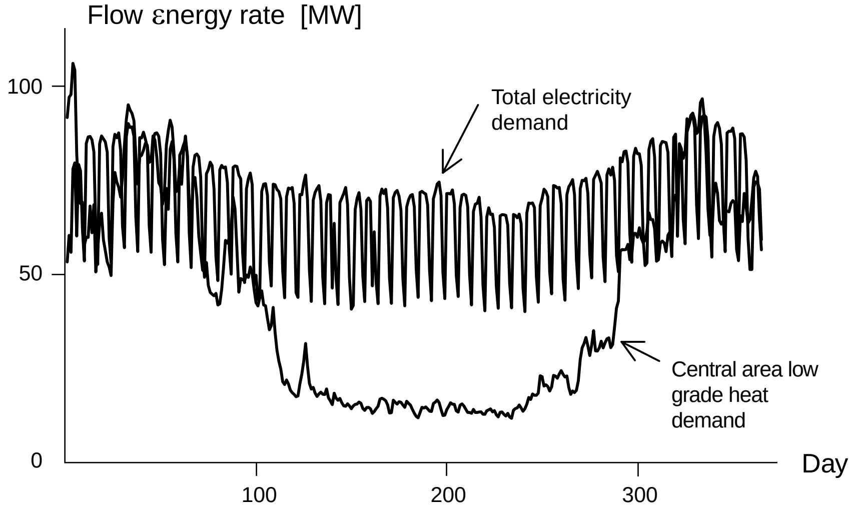
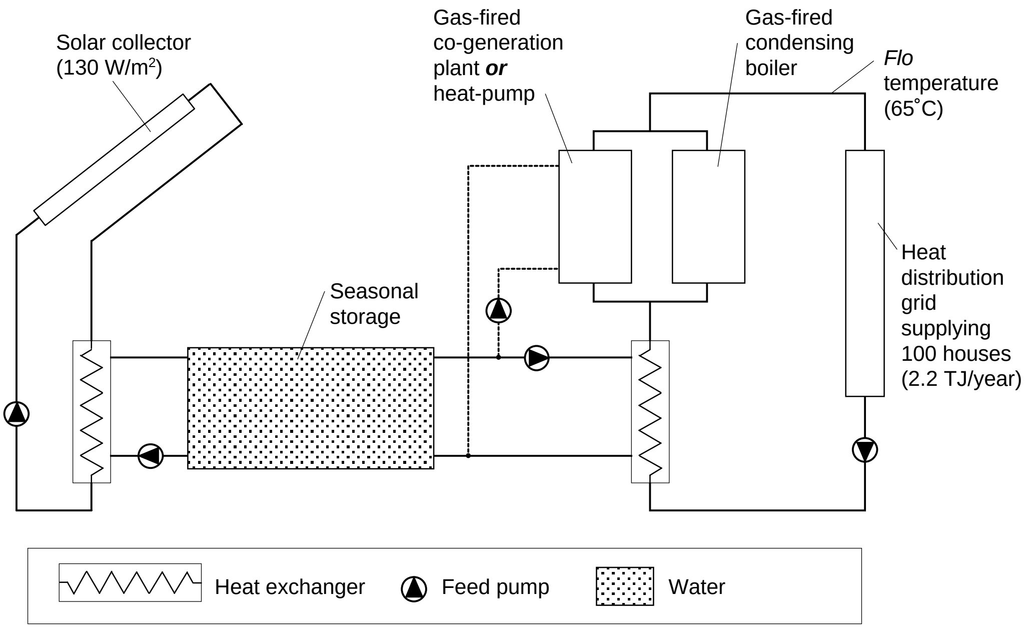


Figure 8.12 *The Würzburg study location-aggregated demand profiles. The space heat demand applies only to those areas potentially able to be supplied by district heating.*[45]

8.13.2 SOLEG solar district heating study

This study uses *deeco* to evaluate solar-recharged seasonal storage district heating options in combination with gas heating, heat-pumps, and co-generation. The system, when finalised, will be used to supply 100 houses in south Germany as part of the publicly-funded SOLEG demonstration project.[46] One arrangement tested is shown in figure 8.13. Results indicate that between 32 and 95% of space heating demand can be supplied by solar insolation rather than purchased exergy.[47] (Lindenberger *et al* 2000)



Notes: 1. The electricity requirements are not shown.
2. A condensing boiler condenses water vapour in the flue gas stream in order to recover the enthalpy of vaporisation (*aka*, latent heat).

Figure 8.13 *Solar-recharged seasonal storage district heating scheme in combinations with various other secondary elements.*

8.14 Closure

This chapter is intended to complement rather than duplicate existing *deeco* documentation. Hence much of the mathematical formalism, and particularly that contained in Bruckner (1997), has been omitted. In addition, the description of

[45] Source: Bruckner *et al* (1997, fig 1, p 1006), re-captioned.

[46] SOLEG is an acronym for solar-supported energy supply for buildings.

[47] Annual average solar insolation in south Germany is 130 W/m^2, compared with 145–180 W/m^2 in New Zealand. Solar insolation is defined on p 176.

deeco given here is framed in terms of graph theory as this provides a useful visual interpretation. Furthermore, it is difficult to conceive of *deeco* being written without the use of graph-based data-types and object-oriented programming — this area of computer science, known as topological computing, is gaining ground, and contemporary graph libraries now bundle sophisticated flow optimisation routines.

deeco is generally better suited to a comparative rather than absolute role. In other words, *deeco* should be used to evaluate network development options by difference with some predetermined reference scenario. Similarly, *deeco* can also be used to identify best-practice benchmarks relative to current performance, again by difference. Furthermore, *deeco* is a decision support tool and is not as a substitute for detailed engineering and cost analysis.

The two studies discussed in section 8.13 are a sample of the type of problems to which *deeco* (and any extensions) could be applied. Another promising application is ′carbon management′ by CO_2 emitting firms who can expect to face emissions pricing in some form in the near future.

The next chapter reviews the option for applying *deeco* to a New Zealand national policy problem. □

Chapter 9

Proposed New Zealand scenario

A spatially-aggregated model of the national electricity and gas systems

This chapter presents a New Zealand exergy sector policy problem to which deeco *could be usefully applied.*

9.1 Introduction

It is the intention of the writer to use *deeco* to analyse a national exergy sector policy problem relevant to New Zealand. This chapter briefly looks at the way such a problem might be framed and how the underlying network model might be structured.

The idea of building a spatially-aggregated model of the New Zealand national electricity and gas systems was first developed with Dr. Groscurth during November 1998. Specifications for new process modules were subsequently outlined and discussed with Dr. Bruckner who provided valuable feedback. The next step was to scope policy problems that would benefit from network optimisation modelling and also to give consideration to the availability of data. In the first instance, the purpose of such modelling would be to demonstrate the potential or otherwise of the *deeco* methodology, rather than to necessarily produce robust policy results — in other words, proof-of-concept could constitute a sufficient goal.

9.2 New Zealand scenario

Issues facing the New Zealand energy sector

The following list of 'problems' facing the New Zealand exergy sector are presented as a scoping exercise without explicit discussion or justification:

- exergy sector responses to the national 0.0% Kyoto Protocol commitment,
- network-effect market failure, particularly that arising from supply or transmission constraint and/or causing unnecessary resource wastage and environmental cost,
- the potential for distributed technologies and issues relating to the stranding of existing plant,
- cessation of production from the Maui natural gas field around 2010,
- dry year hydro-storage management,
- demand management opportunities and uptake.

Modelling strategy

The selected model would need to be indicative, at least in its first phase of development — therefore:

- the model would be highly spatially aggregated, *eg*, South Island hydro-generation would initially be represented by just one scheme,
- time-series should be sourced from original data wherever possible, but, as a fall-back, these could be generated stochastically from low-resolution data,
- data preparation would be via RDBMS (relational database management system) output, which could also provides some pre-*deeco* data integrity checking.

The modelling philosophy would be to start with a relatively simple model in structural terms, and then add complexity as insight and experience build.

New deeco modules required

The following new process modules would be required:

- hydro-scheme module (with storage),
- electricity transmission module (with quadratic losses),
- wind-farm module,
- New Zealand house module,
- wood-burner module,

- natural gas extraction module,
- natural gas transmission module.

Of these, the electricity transmission module presents an interesting problem. Using a DC power flow approximation, transmission losses can be modelled as I^2R (the symbols have their usual meaning) or, in exergy flow terms as $k\dot{E}_{\mathrm{W}}^2$, with k being a constant.[1] These type of quadratic losses can be captured in *deeco* using a piecewise approximation (see p 143). The transmission module input-output equations are probably best derived from engineering design information, rather than constructed from first principles.

Data availability

The problem of data availability is difficult to assess at this point, but could present a substantial problem. The corporatisation, privatisation, division, and rationalisation that has accompanied the exergy sector reforms have led to information being more likely to be regarded as commercially sensitive, fragmented, and/or lost. On the other hand, the various markets now generate considerable data in their own right, some of it hitherto not calculated, and grid operators, in particular, face mandatory reporting under competition law including specific disclosure regulations.

9.3 Closure

Although it is premature to finalise a New Zealand policy scenario that warrants investigation at this stage, formulating possible policy applications for *deeco* is important. It would also appear that the new Labour-led government will take a more inclusive approach to exergy sector public policy formation, and be more proactive in the face of competition failure and social and environmental deficits, than was the previous administration. □

[1] The DC power flow approximation is also used for the nodal pricing of wholesale electricity.

Chapter 10

Conclusions

This chapter collects and summarises the major themes within this thesis.

Optimisation modelling for ′energy′ sustainability

The dominant theme in this thesis is that exergy-services supply systems can and should be systematically integrated to improve their sustainability. Sustainability, in this case, can be interpreted in terms of reduced environmental impact and reduced dependence on depleting resources, including the capacity of the atmosphere to buffer greenhouse gas concentrations.

Integration modelling is best achieved when the network structure — which exists physically in the network architecture, and informationally within the demand profiles and the environmental conditions which prevail — is explicitly included *and* at a sufficiently high spatial and temporal resolution. The less sophisticated ′pool′ approach which has dominated recent energy policy analysis is now severely limited, particularly as emerging distributed technologies begin to erode the incumbent supply-side paradigm. Furthermore, the distributed technologies have attributes which make a well-resolved network architecture, stable storage policies, and active demand management even more important.

′Energy′ resource assessment

This thesis makes use the environment-dependent flow and nonflow exergy functions to represent the flow and stock resources colloquially known as ′energy′. Further, it introduces the environment-dependent flow and nonflow ϵnergy functions, as the first law analogues of flow and nonflow exergy respectively.[1] It also defines the ratio of flow exergy to flow ϵnergy as the ′flow exergy quality factor′ to generalise the concept of fuel and product quality. Exergy quality mismatch then becomes a

[1] Flow and nonflow ϵnergy and exergy are summarised in table 5.3 (see p 56).

indication of thermodynamic waste, and conditions of significant mismatch involving nontrivial exergy flows suggests that network modifications should be explored — such as switching fuels, utilising surplus co-production, adding co-generation, heat-pumps and expanders, or shifting exergy-services demand.

This thesis also examines exergy resources in economic terms and draws a distinction between exergy-services supply systems and $nergy systems.[2] Firms and households demand exergy-service, but such services need not necessarily be supplied through the use of priced $nergy.

The application program deeco

The UNIX-based application program *deeco* provides a modelling environment for seeking better integration of network architecture, plant performance, and demand by time and place in context of the prevailing environmental conditions. *deeco* as a network operation/or and design decision support tool cannot identify efficiency improvements within individual plant, but it can be used to assess whether potential plant efficiency gains would be worth pursuing.

deeco, like most other exergy policy models, is used for comparative analysis, whereby scenario modification sensitivities rather than absolute predictions are sought. One promising role for *deeco* in this context relates to the allocation of net CO_2-e emissions entitlements. Companies which demonstrate that their post-1990 plant and management upgrades have reduced their business-as-usual net CO_2-e output may be eligible for emissions credits. *deeco* could help quantify such gains.

Transferring *deeco* to an Intel-compatible platform made up a substantial part of this thesis. The leaning overhead required to come to terms with language (C++, shell scripting, PERL), development environment (UNIX variants, compiler specifications, system utilities), and porting issues (component libraries, exception handing, floating point arithmetic) was considerable, even though the codebase modifications required to port *deeco* to SCO UnixWare 2.1.2 SDK / x86 turned out to be relatively small in number. Further, *deeco* has limited English language documentation, so that gaining an in-depth understanding of the application itself was arduous.

Closure

It is hoped that this thesis may make a contribution to the exergy-services supply systems sustainability debate in two ways — first, by articulating a network-based view of such systems, and second, by stimulating interest in the application program *deeco*. □

[2] These destinctions are discussed on p 4.

Appendix A

Terminology

This appendix reviews key terms used in this thesis — a task made necessary by virtue of the number of discipline-specific definitions in circulation.

A.1 Introduction

A number of terms have been assigned specialist meanings for the purposes of this thesis — particularly those concerning exergy networks and exergy transactions. This semantic precision incurs some cost, but is necessary to ensure that the network description being developed can be fully generic.

A.2 Defined terminology

The following terminology is provided, noting that these observations are intended to clarify usage rather than provide fully rigorous definitions.

Plant-related terms

It is assumed that the characteristics of any flow or stock commodity can be legitimately captured by a single set of intensive thermodynamic and physical attributes. This proposition, named the ′bulk commodity approximations′ (see p 60), creates a simplified view of thermodynamic plant and plant interconnections, and facilitates the development of the network model at the centre of this thesis. Based on this framework, the following plant-related terms are defined — most are also illustrated (see p 63).

- **plant** are the interconnected discrete entities which, taken collectively, represent the *infrastructure of a given *ESSS. If an exergy system cannot be so captured, then the techniques presented in this thesis are not applicable and a continuum model is indicated. Plant are taken to have a simply identifiable high-level function, and cannot be further reduced whilst still maintaining this characteristic. A plant is surrounding by an implied system boundary which is sited such that all consequent exergy destruction can be assumed to occur within the contained volume.[1]

- **logical plant** is an abstraction used to represent a contiguous group of actual plant as single discrete entity.
- **cyclic plant** makes use of an ideal or compromised engineering cycle for one of three roles: heat-based power production, heat or work-based refrigeration, or heat or work-based heat-pumps. Such devices can be cyclic in either time (*eg*, an internal combustion engine) or space (*eg*, a gas turbine). The term ′cyclic plant′ is adopted in the absence of a better alternative. The optimisation techniques presented in this thesis cannot be applied to the *components which make up cyclic plant. Solving the state of cyclic plant typically requires the use of iterative numerical techniques.
- **components** are the sub-systems within *plant, typically within *cyclic plant, between which *cut-points can be readily inserted. Components are not identified as part of the model under consideration.
- **cut-point** refers to the point where a generalised *commodity flow is ′cut′ for the purposes of intensive variables measurement and flow metering or intensive variable and flow assessment, as appropriate. The term ′cut point′ is adopted in the absence of a better alternative. The location of cut-points defines system boundaries and *vice versa* (see p 61).
- **exergy flows** may be either *actual exergy flows or *logical exergy flows.
- **actual exergy flows** are those *commodity flows entering and exiting a given *plant and valued for their exergy content. It is assumed that any given exergy flow can be exclusively classified as work transfer, mass transfer, or heat transfer. If this restriction is not acceptable then the techniques presented in this thesis are not applicable and a more complex mixed mode treatment would be indicated.
- **logical exergy flows** are used to replace the *actual exergy flows which circulate between two plant with a single interconnecting flow. Logical exergy flows are required in order to implement the flow network model being articulated. The term may also be used to include actual exergy flows which comply with the flow network model requirements without further reinterpretation.

[1] ′Plant′ are known as ′processes′ within the *deeco* documentation.

- **fuel flows** are those *actual or *logical exergy flows necessary as inputs to operate the plant in question. Actual fuel flows may also exit the plant in question, but logical fuel flows always terminate at the plant.
- **product flows** are those *actual or *logical exergy flows, if any, which represent purposeful output. If more than one product flow exists, then these flows are reclassified as co-product flows. Actual product flows may also enter the plant in question, but logical product flows always originate at the plant.
- **waste flows** are those unavoidable *actual or *logical exergy flows not used as fuel flows in downstream processes. Waste flows, by definition, possess exergy. Waste flows can also be viewed as unused co-product flows.
- **extra-flow losses**[2] are those losses that occur as a result of mass transactions at the dead-state chemical potential, heat transactions at ambient temperature T_0, or volume changes at atmospheric pressure P_0 undertaken directly across the system boundary of the plant in question but outside of a cut-point. Furthermore, the exergy destruction associated with these transfers is deemed to take place within this boundary.
- **stream** refers to any mass flow entering or leaving the plant via a port.[3]
- **port** refers to any cut-point involving mass flow.
- **active plant** provides an incremental increase in exergy-service whilst necessarily increasing demanded exergy flow. A resistance heater is an example of active plant.
- **passive plant** provides an incremental increase in exergy-service without increasing demanded exergy flow. Retrofitted insulation is an example of passive plant.
- **SSSF** (steady-state steady flow) *plant are taken to have constant rate and fixed attribute commodity flows entering and exiting via *ports, and constant (including zero) nonflow commodity inventories. A *network containing only SSSF plant is also deemed SSSF.
- **NSSF** (non-steady-state steady flow) *plant are as above, but nonflow commodity inventory may vary, hence internal accumulation and discharge is allowed. A *network containing at least one NSSF plant must itself be regarded as NSSF.

Exergy transactions across a cut-point are labelled as 'flow' transactions, whereas energy transactions which take place across a plant boundary outside of a cut-point

[2] In this context, 'extra' means 'outside of'.

[3] Some authors use the term 'stream' to include heat and work flow in addition to mass flow.

are labelled as ′extra-flow′ transactions — this latter type of transaction is, in the ideal limit, free of exergy.[4] This somewhat cryptic notation is adopted because the overarching network model is framed in terms of inter-plant flows. The terms ′heat′ and ′work′ in the strict sense indicate energy in transfer, but in this thesis, ′heat′, ′heat transfer′, and ′heat flow′ are considered synonyms, as are ′work′, ′work transfer′, and ′work flow′. Actual exergy flows which circulate between neighbouring plant must be reduced to a single logical exergy flow, in order to allow a flow network model to be pursued. Actual exergy flows circulating between groups of components by way of working fluid cannot be directly included in the model, and can only be accommodated by applying a single plant which spans the collection such that the cyclic structure becomes hidden.[5]

Two other terms require specific comment — ′logical′ is used in its computer science sense (see p 175), and ′fuel′ is used in its chemical engineering sense, which is wider than normal usage would suggest (see p 175).

Energy-related terms

The commodity type of interest in this thesis is generic exergy. However, each exergy flow will need to be classified as a different commodity if any or all of its attributes differ. Commodity measures come in two types: flow and nonflow. Flow-based measures are used to describe inter-plant transactions, whilst nonflow measures capture intra-plant storage. The flow-type qualifier can be omitted if implied by the context or if the comment relates to both forms.

Particular semantic difficulties arise when discussing the concept of *energy* from the twin perspectives of applied physics and public policy — namely that the word is definitionally overloaded and cannot be assigned a single unequivocal meaning. The strategy adopted in this work is to typeset the word in three ways, as initially given in table 1.1 (see p 2).

- **energy** (typeset normally) is reserved for the first law meaning of the word. Energy is taken to be a theoretical primitive with contributions from thermodynamics and mechanics — Gibbsian-based thermodynamics provides internal energy U and classical mechanics provides *macroscopic kinetic and potential energy in various forms. These contributions are both additive and are collectively conserved. Statistical physics, in turn, provides more fundamental definitions for internal energy and related state functions.
- **ϵnergy** (or 𝔈nergy) is the potential of a flow or stock *commodity to produce heat, in context of the prevailing environmental conditions. 𝔈nergy

[4] Exergy-free ϵnergy is sometimes termed ′anergy′.

[5] Groups of plant exchanging electricity or mechanical work are legal within *deeco*. This is because electricity and mechanical work cannot vary in terms of exergy quality (see p 65) as they circulate.

is conserved, in the sense that no destruction term exists within an ϵnergy balance. ℑnergy is solely a physical construct and requires no assessment of purpose.

- **exergy** is the potential a flow or stock *commodity to produce work, in context of the prevailing environmental conditions. Exergy is not conserved in real processes, in the sense that a destruction term exists within an exergy balance. But like ϵnergy, exergy is solely a physical construct and requires no assessment of purpose.
- **$nergy** (or $nergy) indicates that the economic meaning is intended, namely a flow *commodity valued for its *exergy content and production convenience, and which cannot be assigned to a higher priced alternative use. $nergy is a multi-dimensional concept and the notion of $nergy therefore depends on the prevailing physical, technical, and economic conditions.
- **macroscopic exergy** indicates those forms of *exergy which can be specifically identified. These include: translational kinetic exergy, rotational kinetic exergy, elastic strain exergy, gravitational potential (GP) exergy, electrical capacitive exergy, and electrical inductive exergy. Macroscopic exergy differs from macroscopic energy in that the inertial frame of reference necessarily coincides with the prevailing dead-state definition.
- **flow** and **nonflow** forms of the ϵnergy and exergy *commodity measures are used for flow and stock commodities respectively.

$nergy and exergy, as defined, are not synonyms. An $nergy resource must be priced, either explicitly by trading or implicitly via opportunity cost — in contrast, an exergy resource can exhibit nil opportunity cost. One such example is solar insolation, which always qualifies as exergy, but depending on circumstances, may not qualify as $nergy. The concept of $nergy is not explicitly required to develop the network optimisation model under consideration, but is necessary when discussing model results from an economic perspective. The plant performance indexes adopted in this work require an assessment of purpose, but this assessment need not be explicitly economic in nature.

Network-related terms

A network view is central to this thesis.

- **commodity** refers to any potential resource which can be described by the time-rate of change of an extensive variable.[6] Commodity measures are

[6] Bruckner (1997, translation) also requires that conservation apply. This restriction is not made here provided the commodity destruction can be specified. Furthermore, the definition given here varies from that used in economics in which a commodity is taken to be a physical factor of production or tangible consumer good which may, in addition, imply relatively little post-extraction processing and relatively high substitutability (Pass, Lowes, and Davies 1988).

defined in two forms — flow and nonflow — which apply to flow and stock commodities, respectively. Flow commodities enter or leave *plant through *cut-points. Stock commodities exist as storage inventory. Waste-heat is classified as a commodity unless it completely lacks *exergy, in which case it would be an *extra-flow loss. The flow of funds can be viewed as a commodity under this definition.

- **biophysical exergy** is that exergy which is extracted directly from the biophysical environment.
- **network** as a consolidation of flow network is used in its mathematical sense. Therefore a network specification requires statements of *infrastructure and location-specific *commodity requests and commodity supply offers. The network is the entity subject to optimisation. In this thesis, it is taken as axiomatic that all requests are met, because the numerical techniques adopted are not able to ration under conditions of scarcity.
- an **exergy-service supply system** (ESSS) is made up of one or more *gatewayed *networks and does not differentiate exergy flows on the basis of opportunity cost.
- **infrastructure** is used to describe the entire physical structure of the *network under consideration, irrespective of jurisdiction or purpose. The consistent identification of network boundaries is an important aspect of the modelling process.
- **reticulation grid** or simply **grid** is used to describe that part of the *infrastructure connecting more than two participants and typically managed under some kind of collective-interest regime.
- **interconnections** join *plant separated by distance and are not necessarily loss-free. Interconnections are directed and may carry *actual exergy flow or *logical exergy flow, as determined by circumstances and interpretation. The network model presented in this thesis requires that any interconnection exhibiting non-trivial exergy destruction be replaced by dedicated transport plant patched in using loss-free *links.[7]
- **links** or **linkages** join actual or *logical plant, are necessarily loss-free, and must be directed and single in the graph theory sense. Conceptually, a link represents the thermodynamic interface between abutting *plant, which may carry either *actual or *logical exergy flow, as appropriate.
- **gateway plant** interface abutting networks and allows for exergy import and export.

[7] 'Transport plant' are known as 'network processes' within *deeco*.

- **demand** indicates demand for *exergy-services (unless the context indicates otherwise).
- ***flo*** and ***return*** (italicised as shown) indicate the circulating transfer medium used in convective heat transfer between two plant.[8] These two actual interconnections need to be represented by a single *logical interconnection within the flow network.

The notion of a reticulation grid is not needed in order to construct an ESSS model, but the concept is useful when discussing network policy issues, including jurisdiction, collective-interest management, and the formulation of network participant incentives and sanctions.

General terms

- **CO_2-e** represents the six greenhouse gases identified by the UN Climate Convention weighted according to their 100-year GWP (global warming potential). The six gases are: CO_2, CH_4, N_2O, NO_x (oxides of nitrogen), CO, and NMVOC (non-methane volatile organic compounds) (Intergovernmental Panel on Climate Change 1994, Ministry of Commerce 1999). In the case of biofuels, sequestered CO_2-e inventory as a result of cropping can be subtracted, given that the cropping regime did not necessitate natural ecosystem conversion. However, any remote joint-implementation greenhouse gas mitigation exercise, including carbon sinking, associated with particular *plant should not be accounted within the optimisation exercise, unless irrevocably linked. Ultimately, decisions relating to CO_2-e inclusion or exclusion reside with the modeller.
- the terms **co-generation** and **CHP** (combined heat and power) are used interchangeably. Some authors restrict co-generation to a particular subset of CHP but no such distinction is made here.
- **efficiency** is used in its physical not economic sense and is taken to mean exergetic efficiency (unless the context indicates otherwise). Exergetic efficiency is defined in eqn (5.15) using average and marginal formulations.
- **ESSS sustainability** and **sustainable ESSS** are imprecise terms, interpreted here as indicating the need to transform the built component of *exergy-services supply system and end-user demand profiles so that *exergy-services are supplied with reduced or zero persistent environmental damage and with markedly less resource depletion, including assimilative capacity, than at present.

[8] The term *'flo'* is spelt in this way to distinguish it from 'flow'.

- **exergy-services** are those services that arise, or could have arisen, from the active consumption of exergy. In practice, the concept is best defined by example, hence refer to the list on p 4. It is assumed in this work that firms and households primarily demand exergy-services, rather than exergy *per se*.
- **fuel**, as general concept, means any exergy plant input, including heat and work, essential for that process.[9] More restrictively, chemical fuel is utilised for its chemical exergy. *Chemical exergy (see p 55) is typically converted into heat or work using a combustor or fuel cell, respectively. A fuel flow need not necessarily exhibit nonzero opportunity cost.
- **logical** is used in its computer science sense, meaning non-physical or conceptual yet underpinned by something physical or actual. For example, logical memory appears contiguous to an application program, but may well be stored on several physical devices (*eg*, RAM and hard-disk) as determined by the operating system.
- **sustainability** is defined in the converse, in which case the concept becomes relative rather than absolute. Hence, sustainability is taken to mean, in operational terms, choosing those behaviours, activities, and technologies which are less, or less likely to be, non-sustainable.[10]
- **thermodynamics** is taken to indicate macroscopic equilibrium thermodynamics plus contributions from classical mechanics needed to include relevant *macroscopic exergy terms (unless the context indicates otherwise).

Economic terms

- an **$nergy supply system** (ES) is that part of an *ESSS which carries *$nergy.
- **consumer $nergy** is defined at the point where a fuel flow is last purchased within the economic system under consideration. Thus consumer *$nergy is an economic concept.
- **primary $nergy** is defined at the point where a fuel flow first enters the economic system under consideration. Thus primary *$nergy is an economic concept.

[9] This usage of the term 'fuel' is adopted from the chemical engineering literature.

[10] Adapted from Peet (1992, p 210–211).

A.3 Glossary

General terms

- ***deeco*** (dynamic energy, emissions, and cost optimisation) is a UNIX-based application program which provides a numerical modelling environment for the systematic integration of network-based exergy-services supply systems. Much of this thesis centres on *deeco* and the ideas and concepts upon which it is based.
- **district heating** (DH) is where multiple dwellings or building are heated using a central source, with hot water or steam typically used as the transfer media.
- **HHV** (higher heating value) is the mass-specific enthalpy of devaluation where water in the flue-gas stream as a result of combustion, and possibly also that originally present in the fuel stream, is condensed. HHV is significantly dependent of the humidity of the combustion air.
- **LHV** (lower heating value) is the mass-specific enthalpy of devaluation where any water in the flue-gas stream remains as vapour. Refer to Nieuwlaar (1996, p 39) for a good description on heating values.
- **NEMESS** (network model of energy-services supply systems) describes the theoretical framework on which *deeco* is based.
- **operational research** (OR) is the study of human-related operational problems, usually by numerical means. Network analysis constitutes a major branch.
- **solar insolation** indicates the strength of both direct and indirect sunlight at the receiving point.

Programming environments, languages, and libraries

- **CPLEX** is a proprietary suite of mathematical programming libraries callable from C++. CPLEX is produced by ILOG.[11]
- **ENERPAC** is a suite for exergy analysis routines, available as ANSI C source code for the cost of documentation (Nieuwlaar 1996). ENERPAC supports several comprehensive environmental reference systems (*ie*, dead-state definitions).
- **FORTRAN** is a high-level non-object-oriented programming language with a long history of use for scientific computation. FORTRAN is typically more portable than C++.

[11] The web URL is: `http://www.cplex.com`.

- **GAMS** (general algebraic modeling system) is a proprietary operational research environment produced by GAMS Development Corporation. GAMS provides an integrated high-level modelling environment for mathematical programming (MP) problems.[12]
- **HTML** (hypertext mark-up language) is used to define the layout of web-pages and to embed links to other WWW resources.
- **LEDA** (a library of efficient data types and algorithms) is a well-developed and documented suite of C++ container class libraries, tested on a number of platforms. The graph library includes optimisation algorithms including MCFP. No payment is required for non-commercial use. See also Mehlhorn and Näher (1999).[13]
- **MATLAB** is a proprietary engineering environment produced by The MathWorks Inc. Computation is based on matrix algebra. Support is provided for linear programming, but not other forms of mathematical programming. An associated product, SIMULINK, provides for dynamic system simulation.[14]
- **MS Access** is a Windows-based relational database system from Microsoft. Programming is by VBA (Visual Basic for applications) scripting and SQL (structured query language) commands.[15]
- **NetClass** is a public-domain network representation and network algorithms library implemented in C++ and favouring STL-like interfaces. Certain OR solvers are being worked on. Philip Walton is developing NetClass as a private initiative.[16]
- **Pascal** is a high-level non-object-oriented programming language. Its usage is declining despite being highly regarded.
- **PERL** is an interpreted language originally developed for processing text files and which now supports a range of Internet protocols.
- **STELLA** is a proprietary graphical-programmed system dynamics environment by High Performance Systems.[17]
- **VBA** (Visual Basis for applications) is used as a scripting language for Microsoft Office applications. VBA is related to Basic.
- **VENSIM** is a proprietary graphical-programmed system dynamics environment by Ventana Systems.[18] □

[12] The web URL is: `http://www.gams.com`.

[13] The web URL is: `http://www.mpi-sb.mpg.de/LEDA`.

[14] The web URL is: `http://www.mathworks.com`.

[15] The web URL is: `http://www.microsoft.com`.

[16] The web URL is: `http://home.att.net/~hpwalton/netclass.html`.

[17] The web URL is: `http://www3.hps-inc.com/edu/stella/stella.htm`.

[18] The web URL is: `http://www.vensim.com`.

Appendix B

Exergy analysis

This appendix presents a reasonably complete treatment of exergy analysis as this material is not available in compact form elsewhere.

The concepts of exergy and exergy destruction need review because the prime function of the systems under consideration is to supply exergy-services (for example, warmth, illumination, transport, or materials processing). Such services necessarily require exergy — sourced either as a primary input or recovered from storage within the network — together with suitable end-use plant.

The concept of exergy was first applied to close-circuit process analysis, but has been extended to cover resource assessment — in which case, the exergy definitions need to be referenced to the prevailing dead-state and not some arbitrarily datum.

B.1 Exergy

This section reviews exergy analysis in general terms, whilst the following two sections provide mathematical expressions for exergy and exergy balance for two classes of problem. Exergy is also covered in the main body of the thesis (see p 58).

> **Energy**: this appendix uses energy rather than ϵnergy when developing analytical expressions, as energy is the more fundamental concept.

B.1.1 Exergy and exergy destruction

Exergy represents the maximum work recoverable from an ′energy′ resource under ambient conditions, in context of known conversion pathways. Exergy destruction arises from the fact that real finite-time processes necessarily create entropy.[1] Entropy generation mechanisms include:

- heat transfer across temperature difference (ΔT),[2]
- thermal (T), (static) pressure (P),[3] or chemical (μ_i) [4] mixing,
- fluid flow with viscous loss (ΔP),
- mechanical friction, electrical resistance, magnetic loss, *etc*,
- unrestrained chemical reaction.

The Gouy-Stodola theorum captures the equivalence between entropy generation and lost work (expressed as rates): [5] [6]

$$\dot{W}_{\text{lost}} = T_0 \dot{S}_{\text{gen}} \tag{B.1}$$

where: $\dot{W}_{\text{lost}}$ is the exergy destruction rate
T_0 is the ambient temperature
$\dot{S}_{\text{gen}}$ is the entropy generation rate

This provides for a rudimentary exergy balance for some arbitrary steady-state process, noting that entropy can only increase:

$$\sum_{\text{in}} \dot{E}^{\star} - \sum_{\text{out}} \dot{E}^{\star} = \dot{W}_{\text{lost}} = T_0 \dot{S}_{\text{gen}} > 0 \tag{B.2}$$

where: $\dot{E}$ is the appropiate flow exergy term (often subscripted)
* indicates the arbitrary datum form is acceptable (see box below)

Notation: E is used to represent the various forms of flow exergy, and $\bar{E}$ to represent total energy.

[1] Entropy can be defined in terms of U and T for a simple system:

$$\left.\frac{\partial S}{\partial U} = \frac{1}{T}\right|_{X} \qquad \text{hence for a quasistatic process:} \quad S_b = S_a + \int_a^b \frac{\text{đ}Q}{T}\Bigg|_{\text{quasistatic}}$$

where: X are remaining extensive state variables, for a fluid system: $X = (V, N_i)^{\top}$
đ indicates an inexact differential (see footnote 25, p 184).

Bejan (1997) eqns 4.20 and 2.38.

[2] Most heat transfer problems reduce to one of minimizing entropy generation: $\dot{S}_{\text{gen}} \approx \dot{Q}\Delta T/T^2$. Such problems split into those which seek exergy transfer by reducing ΔT through augmentation, and those which seek to retain exergy by reducing $\dot{Q}$ through insulation (Bejan 1997, p 137).

[3] Pressure dissimilarity mixing includes the unrestrained expansion of an ideal gas into a vacuum: $S_{\text{gen}} = \frac{P_a V_a}{T_a} \ln \frac{V_a + V_b}{V_a}$, a indicates the gas, and b the vacuum prior to the process (Kotas 1995, p 15).

[4] Species-specific chemical potential μ_i is defined in eqn (B.3).

[5] Bejan (1997) eqn 3.7 and Carrington (1994) eqn 7.2.14.

[6] The dot notation is used to indicate flow — refer to the box on p 57. Arguably, as the terms in this equation are not associated with flows, the d/dt notation would be more appropriate.

Datums: quantities with a $\star$ means an arbitrary but problem consistant environmental datum can be used, whereas quantities without a $\star$ means a suitable ambient environomental datum must be used. In all cases T_0, and P_0 if needed, must be absolute. (Refer to section B.1.3).

Notation: the terms ′flow′ and ′nonflow′ refer to process streams only, whereas ′steady-flow′ and ′non-steady-flow′ refer to the process streams *and* heat and work interactions.

It may be feasible to estimate $\dot{S}_{\mathrm{gen}}$ directly using analytical and empirical arguments applied to the process itself,[7] or, more commonly, $\dot{S}_{\mathrm{gen}}$ can be inferred using an exergy balance similar in form to eqn (B.2).

B.1.2 Exergy analysis

Exergy analysis takes two forms:

- quantifying flow and stock **exergy resources** under prevailing conditions,
- analysing **process performance** at various spatial resolutions.

The two forms given are essentially dual — quantifying exergy resources requires idealised process chains — and process audits require predetermined exergy resource inventories.

Krakow (1991, p 328) aligns with the second when defining exergy analysis as “an implicit comparison of the performance of real thermal systems with the performance of ideal reversible systems.”

The notion of thermal system deserves attention — a ′thermal system′ is taken to be any process or collection of processes which uses heat transfer $\dot{Q}$ as a necessary input, unavoidable pathway, or sought output.

Exergy analysis requires exergy destruction be limited to identified processes and that process interlinking is lossless. Common thermodynamic approximations (such as adiabacity) can be applied to the processes themselves as appropriate.

Exergy analysis is premised on an idealised environment, made up of non-interacting heat, atmospheric, and species-specific mass reservoirs. The question of determining an operational environmental dead-state, when the environment itself is not at

[7] For example, consult: Bejan (1982), Slattery (1981), Sun and Carrington (1991).

equilibrium, is discussed in section B.1.3. Given a suitable dead-state methodology, exergy analysis offers the only theoretically consistent way of comparing resource inventories across space and time.

Exergy resources which reach ambient temperature and pressure equilibrium but without exchanging mass with their environment are classified as ′thermomechanical′ and subscripted $_{\mathrm{x}}$. Resources which also reach chemical equilibrium by exchanging mass are classified as ′total′ and subscripted $_{\mathrm{t}}$ — noting that, in this thesis, the treatment for total exergy *ignores* any kinetic or gravitational potential (GP) exergy in the process streams — hence, total exergy is not a generalisation of thermomechanical exergy.

Exergy analysis can indicate where theoretically avoidable exergy destruction is occuring, and, in this respect, is more informative than an equivalent energy analysis.

Exergy networks

Exergy analysis naturally extends to encompass a network approach comprising interconnected multi-connection plant of the type shown on p 63. Coupling between networked processes, whereby a state change in one process materially influences the state of interconnected processes, generally complicates the analysis. As a result, a number of restrictive conditions may be necessary in order to make this class of problem tractable to flow network optimisation (as described in the body of this thesis).

Treatment

Exergy analysis is based on Gibbsian thermodynamics.[8] The treatment presented here is split into two complementary strands — partly for convenience, because chemically inactive systems are best handled using mass-specific quantities, whereas

[8] The foundation of Gibbsian thermodynamics is the discrete simple system, internally uniform, and characterised by a suitable set of independent thermodynamic state variables. The simple system, together with two of the three following theoretical primitives: energy, entropy, and thermodynamic temperature, gives rise to the system fundamental relations, which can take two equivalent forms (stated for fluid systems): $U(S, V, N_i)$ and $S(U, V, N_i)$. Gibbsian thermodynamics concentrates on reversible processes, brought about by contraint relaxation between simple systems. In most practical cases, the fundamental relation is not explicitly identified, instead some salient feature of this relation is accessed via an empirically-determined response function (*eg*, constant-pressure heat capacity). Consult Callen (1985) or Carrington (1994) for more detail. Gibbsian thermodynamics has been combined with the CKC (Clausius-Kelvin-Caratheodory) formulation and extended to cover cyclic and dissipative processes (engineering thermodynamics), issues of process rate (finite-time thermodynamics), and continuum systems (non-equilibrium thermodynamics (NET)). Note that there is no single-pass development for macroscopic thermodynamics — concepts introduced early on are often reintroduced with more formalism and generality as the theory builds. Statistical mechanics, which incorporates kinetic theory, provides fundamental definitions for energy and entropy at a quantum level.

systems interacting externally, chemically[9] are best handled using molal quantities.[10] These two strands (presented in sections B.2 and B.3) comprise:

- **thermomechanical exergy** conversion processes, where the kinetic and gravitational potential (GP) flow exergies contained within process streams are *accounted for*,[11]
- **total exergy** conversion processes, where the kinetic and gravitational potential flow exergies contained within process streams are *ignored*.

The term ′process stream′ refers to the mass flows entering and leaving the process via indexed ports. Kinetic and gravitational potential flow exergies[12] within the process streams may be tracked separately (as per Kotas[13]), or included within the definition of thermomechanical exergy (as per Bejan) — the latter approach is adopted here.

Process classification

The processes under consideration are characterised as ′non-steady-state steady-flow′, abbreviated as *NSSF*, or some reduction of this case. NSSF means that all process streams and heat and work interactions are required to be time-invariant, but that the state of the process in bulk terms (*ie*, the mass-averaged thermodynamic state variables) may be time-variant. *Steady-state*[14] means that the process state, excluding entropy S, must remain invariant in bulk terms.

In all cases, the environmental intensities T_0, P_0, and $\mu_{0,i}$ are taken to be constant, as are all external parameters which may influence the process, such as gravitational, electric, and magnetic field strengths.

[9] Such interaction does not need to be chemically reactive — diffusive contact involving one component is sufficient to classify the system as externally chemically active.

[10] If X represents an extensive property of a simple system, its proper and partial molal values are, respectively:

$$\bar{x} = \frac{X}{N} \qquad \text{and} \qquad \bar{x}_i = \left. \frac{\partial X}{\partial N_i} \right|_{T,P,N_{j \neq i}}$$

See Bejan (1997) eqns 4.107 and 4.101. Mass-specific quantities can be converted to molal quantities by dividing by the molecular weight (also known as molar mass) of the system or species concerned (in g/mol) and multiplying by 1000 g/kg. Some authors use the term ′molar′ to mean proper molal. Do not confuse this concept with that of mole fraction x_i defined in eqn (B.31).

[11] Thermomechanical exergy is sometimes called physical exergy. It can also be subscripted $_{\mathrm{tm}}$.

[12] Kinetic and gravitational potential exergies are indentical to their more usual energy counterparts, except that they are referenced to an appropriate ambient enviromental dead-state rather than some arbitrary datum.

[13] Hence, Kotas (1995) eqn 2.4 defines total flow exergy thus:

$$\dot{E}_{\mathrm{t}} = \dot{E}_{\mathrm{k}} + \dot{E}_{\mathrm{p}} + \dot{E}_{\mathrm{x}} + \dot{E}_{\mathrm{ch}}$$

where: $_{\mathrm{k}}$ indicates kinetic exergy, and $_{\mathrm{p}}$ potential exergy, including gravitational potential.

[14] The term ′stationary′ can be regarded as synonomous with steady-state in this context.

Nonflow (*ie*, batch) processes[15] are a particular case of NSSF whereby process streams are absent, but cross-boundary heat and work interactions may still occur.[16] Conversely, *SSSF* or ′steady-state steady-flow′ (*ie*, continuous) processes are a particular case whereby nonflow exergy Ξ and system mass M accumulaton is nil.

Many engineering applications involve cyclic processes, either linked in time and requiring piecewise nonflow analysis (*eg*, Stirling cycle), or linked in space and requiring multi-component SSSF exergy analysis (*eg*, Rankine cycle). In addition, SSSF analysis can be applied to systems operating cyclically in time given their period-averaged behaviour is stable, and smoothed results are acceptable.

Process aggregation and process grouping

Exergy is as much a network concept as it is a process concept. No useful process can exist in isolation — all processes ultimately require exergy input from upstream processes, and all are influenced by downstream events and by ambient conditions.

Questions then arise as to where external boundaries should be set and what level of process aggregation is necessary when modelling some system of interest — these can only be resolved by the modeller, contingent on purpose, data availability, and project resourcing. Notwithstanding, exergy analysis allows coarsely modelled processes to be dissaggregated at some later point, and groups of processes to be collapsed into one control region if need be.[17]

Hence, although exergy is not a conserved quantity, exergy can be audited at any chosen level of resolution, such that the results remain externally consistent.[18]

However, disaggregated processes may or may not possess stand-alone functionality. The term ′plant′ is taken to be the minimum collection of processes which together perform some desired external function. In effect, plant represents the least unit to which a first law efficiency measure could be usefully applied. All engineering-cycles (cyclic in time or space) therefore qualify as plant.

The term ′closed-circuit′ indicates that an engineering-cycle working substance does not come in direct contact with the environment.

[15] By the scheme in use, these could be tagged ′NSNF′ — non-steady-state non-flow.

[16] Such systems are often termed ′closed′, but care is needed as no consistent definition for closed system is in use — Bejan (1997, p 3) allows energy but not mass exchange, whereas Kittel and Kroemer (1980, p 29) require constant values for all conserved extensive variables (such as energy, particle count, and volume) and all external parameters that may influence the system (including external field strengths such as g). Hence the term ′closed system′ is avoided in this work.

[17] Matrix-based algorithmic graph theory can provide a convenient method of aggregating and disaggregating such processes. This strategy is not adopted in *deeco*, but it has been successfully coded in FORTRAN 77 for real-time power plant exergy auditing (Yasni and Carrington 1988).

[18] A more formal statement of this idea is contained in Kotas (1995) eqn 3.42, whereby the δ_i can be aggregated linearly, δ_i being the component efficiency defects.

The bulk commodity approximations

The ′bulk-flow′ and ′bulk-storage′ approximations are given elsewhere (see p 60). But to recap, the bulk-flow approximation for flow processes requires that the following conditions hold at all inlet and outlet ports:[19]

- bulk-flow mechanical conditions apply,
- port surfaces can be considered single simple systems and thus in local thermodynamic equilibrium (LTE).[20]

The first point requires that process streams can be validly represented (as required) by: flow-rate $\dot{m}$ or $\dot{N}$, bulk velocity υ,[21] and bulk altitude z. The second requires that the port surface can be represented by one set of thermodynamic intensities (as required): temperature T, (static) pressure P, and chemical potential μ_i.

The bulk-flow assumption is analogous to the equilibrium endpoint requirement for the analysis of nonflow processes.

Chemical potential

The chemical potential for species i for a fluid system can be defined by:[22] [23]

$$\mu_i = \frac{\partial G}{\partial N_i}\bigg|_{T,P,N_{j\neq i}} \tag{B.3}$$

Hence, the chemical potential μ_i can be regarded as shorthand for the partial molal Gibbs free energy of some particular species i reversibly entering or leaving a system held at constant T and P (but not necessarily T_0 and P_0) via a semi-permeable boundary:[24]

$$\mu_i = \bar{g}_i \bigg|_{T,P,N_{j\neq i}} = \left(\bar{h}_i - T\bar{s}_i\right)\bigg|_{T,P,N_{j\neq i}} \tag{B.4}$$

The (differential) chemical work is then given by:[25]

$$-\text{đ}W_{\text{chem}} = \mu_i \, dN_i \bigg|_{\substack{\text{reversible}\\ T,P,N_{j\neq i}}} \tag{B.5}$$

[19] Bejan (1997), p 70.

[20] Simple systems are LTE by definition. Non-equilibrium thermodynamics (NET) is premised on a continuum of simple systems.

[21] Using RMS (root-mean-square) velocity sampled across the port, or by accepting plug flow so that $\upsilon = \dot{m}\rho/\text{CSA}$, where CSA is the cross-sectional area [m^3] and ρ is mass density [kg/m^3].

[22] An alternative equivalent definition for the chemical potential is:

$$\mu_i = \frac{\partial U}{\partial N_i}\bigg|_{S,V,N_{j\neq i}}$$

[23] For a single species system, simply use μ.

[24] The term ′species-specific selectively-permeable boundary′ would make a better description.

[25] The inexact differential đ indicates the result requires knowledge of the process path. Kotas (1995, p xvii) describes the difference thus: d represents an infinitesimal change in a state variable, đ represents an infinitesimal transfer by some mechanism.

The *dead-state* chemical potential for species i (*ie*, of the mass reservoir) $\mu_{0,i}$ interacting through a semi-permeable boundary is equal to the partial molal dead-state Gibbs free energy for that species. Hence, by a reduction of eqn (B.4):[26]

$$\mu_{0,i} = \bar{g}_{0,i} = \bar{h}_{0,i} - T_0 \bar{s}_{0,i} \tag{B.6}$$

Unlike T and P, μ_i cannot be measured directly. However, $\mu_{0,i}$ can be determined, with knowledge of $\bar{h}_{0,i}$ and $\bar{s}_{0,i}$, using eqn (B.6).

B.1.3 Environmental dead-state

The environmental dead-state is a defining concept for exergy analysis. A dead-state is specified using the principle that the local environment should not be a source of energy convertable to work. The term 'local' is inserted to indicate that the choice of dead-state is problem-specific and typically requires some judgement of the part of the modeller.

Environmental state and mechanics variables

In all cases, knowledge of T_0 is required. But, depending on the type of environmental interactions at play, further dead-state definitions may not be necessary.

A real *or* hypothetically-ideal plant may exchange the following with the environment:

- entropy at T_0 (*ie*, reject heat[27]),
- volume at P_0 (*ie*, atmospheric work),
- substance (moles of) at $\mu_{0,i} \quad i = 1 \ldots n$ (*ie*, chemical work).

A real *or* hypothetically-ideal plant may also exchange (*ie*, appropriate or credit) kinetic and gravitational potential (GP) energy with its surroundings as work transactions:

- kinetic energy, with velocity $\upsilon - \upsilon_0$ relative to the earth's surface (normally an inertial frame is selected such that $\upsilon_0 = 0$, but this is not mandatory),
- gravitational potential energy, with height $z - z_0$ relative to ground-level.

A dead-state definition, beyond knowledge of T_0, need only encompass the thermodynamics intensities P_0, $\mu_{0,i}$ and mechanics variables υ_0, z_0 with which the plant directly interacts. At this point, two key issues arise:

- the choice of datums for thermodynamic and mechanics variables,
- the selection of a suitable enviromental dead-state.

[26] Kotas (1995) uses the notation μ_{i0} instead.

[27] Reject heat is sometimes known as 'anergy' transfer — transfer of energy without exergy.

Datums for variables

Before state properties and mechanics variables — such as u, h, s, υ and z as may be needed for the analysis — can be given numerical values, suitable problem-constistant variable datums and an environmental dead-state must be specified.

Thus, taking y to represent any such variable or derived variable (with the notation $_0$ is used to indicate the variable exhibits its dead-state value):

$$y \in \{h, s, z, \upsilon \ldots\} \cup \{g, e_\mathrm{x}, \bar{e}_\mathrm{t} \ldots\} \tag{B.7}$$

Two strategies present for managing a non-arbitrary datum system:

- setting the $y_0 = 0$ by specifying suitable problem-specific datums,
- allowing the y_0 to float, and including these terms in all relevant algebraic expressions, thereby allowing the y to use arbitrary datums.

In this thesis, the second more general strategy is adopted. However, in some equations, most notably some balance equations, arbitrary datums (excepting T_0, and P_0 if required) are acceptable.

Hence, the following relationship holds for the y:

$$y_{\substack{\text{referenced}\\\text{to ambient}}} = \overset{\star}{y} - \overset{\star}{y}_0 \tag{B.8}$$

where: $_0$ indicates the dead-state value

Exergy for which an arbitrary datum is acceptable (but not mandatory) is indicated in this thesis using a $\star$, for example:[28]

$$\overset{\star}{\Xi}_\mathrm{x} \qquad \overset{\star}{\Xi}_\mathrm{t} \qquad \overset{\star}{e}_\mathrm{x} \qquad \overset{\star}{\bar{e}}_\mathrm{t}$$

where: $_\mathrm{x}$ indicates thermomechanical exergy
$_\mathrm{t}$ indicates total exergy

The relationship between ambient (*ie*, non-arbitrary) and arbitrary datummed exergies is therefore:

$$\begin{aligned}
\Xi_\mathrm{x} &= \overset{\star}{\Xi}_\mathrm{x} - \overset{\star}{\Xi}_\mathrm{x,0} \\
\Xi_\mathrm{t} &= \overset{\star}{\Xi}_\mathrm{t} - \overset{\star}{\Xi}_\mathrm{t,0} \\
e_\mathrm{x} &= \overset{\star}{e}_\mathrm{x} - \overset{\star}{e}_\mathrm{x,0} \\
\bar{e}_\mathrm{t} &= \overset{\star}{\bar{e}}_\mathrm{t} - \overset{\star}{\bar{e}}_\mathrm{t,0}
\end{aligned}$$

Note that exergy (in its various forms) is the only variable to which the $\star$ notation is *consistently* applied in this appendix. Although all other variables, excepting T and

[28] Kotas (1995, p 256) also introduces the $\star$ form in a similar context.

P, are defined on arbitrary datums, the $\star$ notation is typically omitted for reasons of compactness — nevertheless eqn (B.8) stands.

Finally, in nearly all cases, an inertial frame is choosen such that $\upsilon_0 = 0$, *ie*, the frame is fixed to the earth's surface.

Some thermodynamic calculations are *independent* of choice of datums and reference system, including some encountered in energy and exergy analysis — for example, process heat loss and process exergy destruction — these both being determined by difference.

Other calculations are *dependent* on choice of datum and reference state — for example, the exergy content in an exiting process stream. In such cases, all relevant extensive thermodynamic variables require an ambient (*ie*, dead-state) definition.

Conversely, the intensive thermodynamic variables T and P must be absolute, based on absolute zero and a vacuum respectively. Furthermore, T_0 must be specified for all problems, and P_0 for those problems involving changes in process volume V.

Environmental reference systems

Nieuwlaar (1996, p 25) identifies two reference systems used in physical chemistry to quantify system exchanging mass with their environment:

- thermodynamic reference system (TRS),
- environmental reference system (ERS).

Only the second is appropriate for exergy analysis.

An ERS is established by using the principle that the environment itself should not be a source of energy convertable to work. Determining a suitable ERS — also denoted as the dead-state definition — is conceptually seperate from the issue of defining datums for variables. However, as noted previously, the two issues are often rendered indistinguishable by setting all $y_0 = 0$ for the problem at hand.

In the ERS, the enthalpy and entropy of stable compounds at T_0 and P_0 are selected such that $h - h_0 = 0$ and $s - s_0 = 0$ for each compound. Other thermodynamic potentials, plus flow and nonflow exergy can be derived using these quantities together with T_0, and if needed P_0, and corrected for concentration (chemical mixing) as necessary.

This system is quite different from the thermodynamic reference systems (TRS) commonly used in analytical chemistry. In this case, $h = 0$ is set for each element at $T = 293.15\,\mathrm{K}$ and $P = 100\,\mathrm{kPa}$,[29] and $s = 0$ at $T = 0\,\mathrm{K}$. Notwithstanding, TRS values are used to calculate ERS values.

[29] The use of 101 kPa has been replaced by 100 kPa in some cases (Aylward and Findlay 1994).

The selection of stable compounds is not straight forward and several schemes are in use. Further details are not warranted in this thesis, but can be found in: Kotas (1995), Krakow (1991), Nieuwlaar (1996).

Operationalising a dead-state for a given problem is not necessarily straight forward. Variations in dead-state conditions across time may present analytical difficulties — Krakow (1994, p 4) defines an 'effective reservoir temperature' and an 'effective reservoir relative humidity' which may assist in such circumstances.

Restricted dead-state and chemical exergy

As noted, a flow or stock exergy resource may exchange the following with the environment in order to produce potential work:

- energy (*ie*, heat and work),
- mass.

In many problems, process streams can never arrive at full dead-state often because of design or circumstances, such as physical containment in closed-circuit plant). In such cases, a restricted dead-state definition is used, and the exergy inventory is thermomechanical. If not, the exergy inventory is total. The difference in exergy inventory for a compound in its restricted and full dead-states, if any, is termed 'chemical' exergy. Hence: [30]

$$E_{\mathrm{t}} = E_{\mathrm{x}} + E_{\mathrm{ch}} \tag{B.9}$$

$$\Xi_{\mathrm{t}} = \Xi_{\mathrm{x}} + \Xi_{\mathrm{ch}} \tag{B.10}$$

where: $_{\mathrm{t,\,x,\,ch}}$ indicate total, thermomechanical, and chemical exergy

In the case of a homogeneous fluid, thermomechanical exergy can be broken down into a thermal component ($e_{\mathrm{x}}^{\Delta T}$) represented by a reversible isobaric process, and a pressure component ($e_{\mathrm{x}}^{\Delta P}$) represented by a reversible isothermal process (Kotas 1995, p 40).

Availability

The concept of availability is frequently encountered in the literature. The relationship between exergy and availability can be given by (noting that this definition of availability is *not* universal): [31]

$$\Xi_{\mathrm{x}} = A - A_0 \tag{B.11}$$

$$e_{\mathrm{x}} = b - b_0 \tag{B.12}$$

[30] Equations (B.9) and (B.10) require consistancy in the formulation of the $_{\mathrm{x}}$ and $_{\mathrm{t}}$ forms.

[31] Bejan (1997) eqns 3.32, 3.43, Carrington (1994) eqn 7.3.24, and Kotas (1995) eqn 2.11.

where:	A	is nonflow availability
	b	is mass-specific flow availability
and:	$_0$	indicates the dead-state value

Alternatively, this relationship can be given using the $\star$ form of thermomechancial exergy:

$$\Xi_{\mathrm{x}}^{\star} = A \tag{B.13}$$

$$e_{\mathrm{x}}^{\star} = b \tag{B.14}$$

Hence, availability is taken to represent the maximum work ideally possible from a given *thermomechanical* conversion process. Hence, availability cannot be applied to a generalised process stream or exergy stock resource, nor to a chemically active process — availability is therefore not as expansive a concept as exergy.[32]

B.1.4 Analytical simplifications

The general cases given in sections B.2 and B.3 may simplify considerably in the following situations:

- processes (and groups of process) with SSSF characteristics,
- nonflow processes,
- systems with process streams that do not interact directly (*ie*, exchange mass) with their environment (*eg*, closed-circuit plant),
- processes whereby streams do not mix internally (*eg*, heat-exchangers),
- analyses whereby at least some environmental datums can be set to zero by careful choice of reference frame.

For the case of SSSF (*ie*, $d\Xi_{\mathrm{x|t}}/dt = 0$) and the exergy balance simplifies considerably. For closed-circuit plant, the environmental datum for the working substance may be arbitrary rather than dead-state, although T_0 must be known. In such cases, the exergy efficiency for such a system and its component processes is hence independent of the prevailing dead-state definition beyond T_0.[33]

B.1.5 Numerical calculation and validation

Various authors advocate that exergy analysis be used numerically rather than descriptively. Krakow (1991, p 335) offers methods for checking numerical consistancy, whilst noting that no verification method can be fully robust.

[32] To reduce confusion, the term 'availability' is avoided in this thesis.

[33] Keenan (1932) presents an steam-turbine exergy analysis conducted under these conditions.

Listings of $\bar{h}^\circ$ and $\bar{e}_{\mathrm{t}}^\circ$ are published, where ° indicates a standardised value requiring adjustment for ambient T and possibly P (in this case, $\bar{h}^\circ$ should not be confused with 'methalpy'[34]). Numerical routines are also available, for example, ENERPAC 5 (Nieuwlaar 1996).

B.1.6 The exergy balance equations

The following sections B.2 and B.3 present expressions for thermomechanical and total exergy balance and their respective flow and nonflow exergies treatment is based on Bejan (1997).[35] [36] No derivations are given, but exergy balances and consequent exergy definitions can be found at by taking mass, energy, and entropy balances for the system under consideration, and solving for:

$$\dot{E}_W = \dot{W} - P_0 \frac{dV}{dt} \tag{B.15}$$

where: $\dot{E}_W$ is the exergy exported as work
$\dot{W}$ is the total work generated

whilst eliminating $\dot{Q}_0$.[37] The appropriate mass, energy, and entropy balances are reproduced toward the end of each section.

Sign convention: an engineering sign convention is used: *heat-in* and *work-out* both $+ve$.

Specific quantities: extensive quantities typeset as lower case are *mass-specific*, and with an over-bar are *molal* (see footnote 10, p 182).

[34] See footnote 43, p 192.

[35] The notation used is similar to Bejan (1997), but changes include:

- cross-boundary indexing for the thermomechanical exergy balance (eqn (B.16)) and related balances is now consistant with the indexing for the total exergy balance (eqn (B.29)),
- Ξ_{x}, rather than simply Ξ, is used for thermomechanical nonflow exergy,
- υ, rather than V, is used for bulk velocity,
- $\star$ indicates arbitrary datum form for exergy, namely: $\Xi_{\mathrm{x}}^{\star}$, $\Xi_{\mathrm{t}}^{\star}$, $e_{\mathrm{x}}^{\star}$, and $\bar{e}_{\mathrm{t}}^{\star}$,
- đ, rather than δ, is used to indicate an inexact differential.

[36] Equations cross-reference with Bejan (1997) as follows:

(B.16)	(B.17)	(B.20)	(B.22)	(B.27)	(B.28)
(3.9)	(3.32)	(3.43)	(3.14)	(3.41)	(3.44)

(B.29)	(B.30)	(B.32)
(5.30)	(5.9′)	(5.22′)

[37] The thermodynamic balances used are not the most mathematically general, but they are appropriate for the purpose at hand. Consult Slattery (1981) for more general vector calculus formulations of these balance equations.

B.2 Thermomechanical NSSF exergy process balance (including stream kinetic and GP exergy)

The exergy balance, in arbitrary datum form, for a generalised non-steady-state steady-flow (NSSF) process (see fig B.1), exchanging entropy at T_0 (*ie*, heat) and volume at P_0 with its environment[38] is given by:[39] [40] [41]

$$\dot{E}_W = -\frac{d\Xi^\star_{\mathrm{x}}}{dt} + \sum_{\ell=1}^{p}(\dot{E}_Q)_\ell + \sum_{j=1}^{q}(\dot{m}e^\star_{\mathrm{x}})_j - \sum_{k=1}^{r}(\dot{m}e^\star_{\mathrm{x}})_k - T_0\dot{S}_{\mathrm{gen}} \tag{B.16}$$

where the following shorthand applies:[42]

$$(\dot{m}e^\star_{\mathrm{x}})_j \equiv \dot{m}_j(e^\star_{\mathrm{x}})_j \quad etc$$

and: $_{\mathrm{x}}$ indicates thermomechanical exergy
* indicates arbitrary datum form is acceptable

also: heat-in and work-out both $+ve$ (engineering convention)

As noted earlier, this exergy balance is best formulated using extensive properties expressed in mass-specific form.

For completeness, mass, energy, and entropy balances for the process are recorded in equations (B.24) – (B.26) respectively.

B.2.1 Thermomechanical nonflow exergy

The thermomechanical nonflow exergy inventory Ξ_{x} of an exergy stock is given by:

$$\Xi_{\mathrm{x}} = \bar{E} - \bar{E}_0 \; + \; P_0(V - V_0) \; - \; T_0(S - S_0) \tag{B.17}$$

where: $_0$ indicates the dead-state value

The total energy $\bar{E}$ is given by:

$$\bar{E} = U + \frac{I\omega^2}{2} + mgz \; + \; \text{other forms of macroscopic exergy inventory} \tag{B.18}$$

Equation (B.16) only requires changes in Ξ_{x} to be accounted for, hence eqn (B.17) can be written in arbitrary datum form:

$$\Xi^\star_{\mathrm{x}} = \bar{E}^\star + \; P_0V^\star - \; T_0S^\star \tag{B.19}$$

[38] Transactions with reservoirs are deemed reversible — hence, all entropy generation is considered to take place within the control region.

[39] Much of this treatment derives from Bejan (1997) chapter 3. See also Kotas (1995) appendix B for a similar derivation.

[40] The dead-state mass flow-rate is assumed to be zero: $(\dot{m}_j)_0 = (\dot{m}_k)_0 = 0$.

[41] As discussed earlier, it is possibly better to write $\dot{S}_{\mathrm{gen}}$ as $\frac{d\dot{S}_{\mathrm{gen}}}{dt}$ but convention precludes this.

[42] See p 186 for a discussion on the * form.

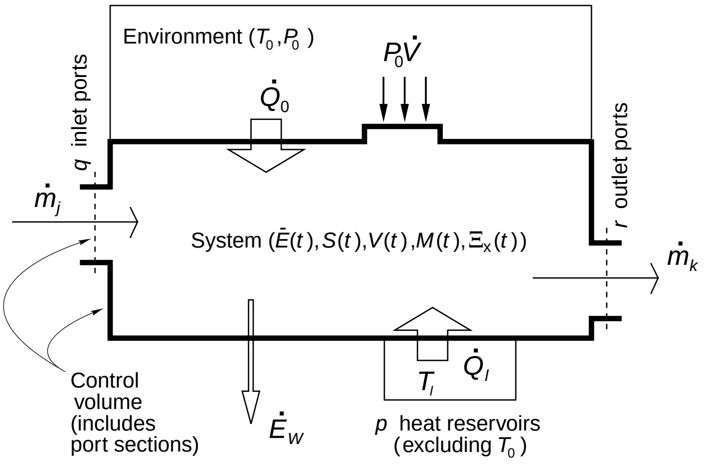

Environment (T_0,P_0)
$P_0\dot{V}$
$\dot{Q}_0$
q inlet ports
$\dot{m}_j$
System ($\bar{E}(t)$,$S(t)$,$V(t)$,$M(t)$,$\Xi_x(t)$)
r outlet ports
$\dot{m}_k$
Control volume (includes port sections)
$\dot{E}_W$
T_l
$\dot{Q}_l$
p heat reservoirs (excluding T_0)

Figure B.1 *A generalised non-steady-state steady-flow process (NSSF) based on thermomechanical exergy and including kinetic and gravitational potential exergies in the process streams.*

Furthermore, if the arbitrary datums coincide with the environmental datums, $\bar{E}_0 = V_0 = S_0 = 0$, then eqn (B.17) reduces to eqn (B.19).

The $I\omega^2/2$ term (assuming: $\omega_0 = 0$) arising from eqns (B.17) and (B.18) refers to rotational kinetic exergy contained within rotating machinery.

B.2.2 Thermomechanical flow exergy

The mass-specific thermomechanical flow exergy e_{x} of a process stream is defined by:

$$\begin{aligned} e_{\mathrm{x}} &= (h + \frac{\upsilon^2}{2} + gz) - (h + \frac{\upsilon^2}{2} + gz)_0 - T_0(s - s_0) \qquad [43] \qquad \text{(B.20)} \\ &= (h - h_0) + \frac{(\upsilon^2 - \upsilon_0^2)}{2} + g(z - z_0) - T_0(s - s_0) \end{aligned}$$

where: $h = u + Pv$ is mass-specific enthalpy

This equation therefore needs to be evaluated at each inlet port $j = 1 \ldots q$ and outlet port $k = 1 \ldots r$ cross-section.

As with nonflow exergy, the arbitrary datum form of eqn (B.20) is encountered:

$$e_{\mathrm{x}}^{\star} = h^{\star} + \frac{\upsilon^{\star 2}}{2} + gz^{\star} - T_0 s^{\star} \qquad \text{(B.21)}$$

[43] Bejan (1997, p 23) introduces mass-specific 'methalpy' h° to simplify this equation:

$$h^\circ = h + \frac{\upsilon^2}{2} + gz$$

As discussed in footnote 13 (see p 182), Kotas (1995) instead accounts for kinetic and gravitational potential flow exergies as distinct entries.

Furthermore, if $h_0 = \upsilon_0 = z_0 = s_0 = 0$, eqn (B.20) reduces to eqn (B.21).

B.2.3 Non-stream heat transfer exergy

The exergy associated with heat transfer from $\ell = 1 \ldots p$ external heat reservoirs is given by:[44] [45]

$$(\dot{E}_Q)_\ell = \left(1 - \frac{T_0}{T_\ell}\right) \dot{Q}_\ell \tag{B.22}$$

B.2.4 Restricted dead-state definition

A restricted dead-state definition requires the following information in appropriate form:

$$T_0,\ P_0,\ \upsilon_0,\ z_0$$

The restricted dead-state chemical potential μ_i^* is dependent on T_0, P_0:

$$\mu_i^* = \mu_i(T_0, P_0) \tag{B.23}$$

where: * indicates the restricted dead-state

B.2.5 Mass, energy, and entropy balances

The mass balance given by:[46]

$$\frac{dM}{dt} = \sum_{j=1}^{q} \dot{m}_j - \sum_{k=1}^{r} \dot{m}_k \tag{B.24}$$

where: M is the mass inventory of the process

The energy balance in arbitrary datum form is given by:

$$\frac{d\bar{E}^\star}{dt} = \sum_{\ell=0}^{p} \dot{Q}_\ell - \dot{W} + \sum_{j=1}^{q} (\dot{m}(h + \frac{\upsilon^2}{2} + gz)^\star)_j - \sum_{k=1}^{r} (\dot{m}(h + \frac{\upsilon^2}{2} + gz)^\star)_k \tag{B.25}$$

where: * indicates the arbitrary datum form is acceptable

The entropy balance in arbitrary datum form is given by:

$$\dot{S}_{\text{gen}} = \frac{dS^\star}{dt} - \sum_{\ell=0}^{p} \frac{\dot{Q}_\ell}{T_\ell} - \sum_{j=1}^{q} (\dot{m}s^\star)_j + \sum_{k=1}^{r} (\dot{m}s^\star)_k > 0 \tag{B.26}$$

where: * indicates the arbitrary datum form is acceptable

[44] Equation (B.22) can be derived by undertaking energy and entropy balances on a Carnot cycle. Kotas (1995) eqn 2.2 defines a dimensionless exergetic temperature τ thus:

$$\tau = 1 - \frac{T_0}{T}$$

[45] Separately identified work transfers could also have been included (refer figure B.1), but were not for reasons of simplicity. In this sense, $\dot{E}_W$ is the *net* useful work rate.

[46] The dead-state mass flow-rate is assumed to be zero: $(\dot{m}_j)_0 = (\dot{m}_k)_0 = 0$.

B.2.6 Simplifications

For a SSSF process with no mixing of streams, comprising $k = 1 \ldots r$ different process streams (*eg*, a heat-exchanger), eqn (B.16) reduces to:

$$\dot{E}_W = \sum_{\ell=1}^{p} (\dot{E}_Q)_\ell + \sum_{k=1}^{r} [(\dot{m}e_{\mathrm{x}}^{\star})_{\mathrm{in}} - (\dot{m}e_{\mathrm{x}}^{\star})_{\mathrm{out}}]_k - T_0 \dot{S}_{\mathrm{gen}} \tag{B.27}$$

If all process streams comprise the same substance, eqn (B.27) further reduces to:

$$\dot{E}_W = \sum_{\ell=1}^{p} (\dot{E}_Q)_\ell + \sum_{\mathrm{in}} \dot{m}e_{\mathrm{x}}^{\star} - \sum_{\mathrm{out}} \dot{m}e_{\mathrm{x}}^{\star} - T_0 \dot{S}_{\mathrm{gen}} \tag{B.28}$$

B.2.7 Closure

Common closed-circuit working substances have published values for h and s as functions of the measureable intensities T and P.

B.3 Total NSSF exergy process balance (excluding stream kinetic and GP exergy)

The exergy balance for a generalised non-steady-state steady-flow (NSSF) process (see fig B.2), exchanging entropy at T_0 (*ie*, heat), volume at P_0, and $i = 1 \ldots n$ chemical species with its environment, as the case may be, is given by:[47]

$$\dot{E}_W = -\frac{d\Xi_{\mathrm{t}}}{dt} + \sum_{\ell=1}^{p} (\dot{E}_Q)_\ell + \sum_{j=1}^{q} (\dot{N}\bar{e}_{\mathrm{t}})_j - \sum_{k=1}^{r} (\dot{N}\bar{e}_{\mathrm{t}})_k - T_0 \dot{S}_{\mathrm{gen}} \tag{B.29}$$

where the following shorthand applies:

$$(\dot{N}\bar{e}_{\mathrm{t}})_j \equiv \dot{N}_j (\bar{e}_{\mathrm{t}})_j \quad etc$$

and: $_{\mathrm{t}}$ indicates total exergy, *ie*, thermomechanical and chemical exergy
$\bar{\ }$ indicates (proper or partial) molal (per mol) quantity

with: heat-in and work-out both $+ve$ (engineering convention)

As noted earlier, this exergy balance is best formulated using extensive properties expressed in molal form (defined in footnote 10, p 182).

[47] Much of this treatment derives from Bejan (1997) chapter 5.

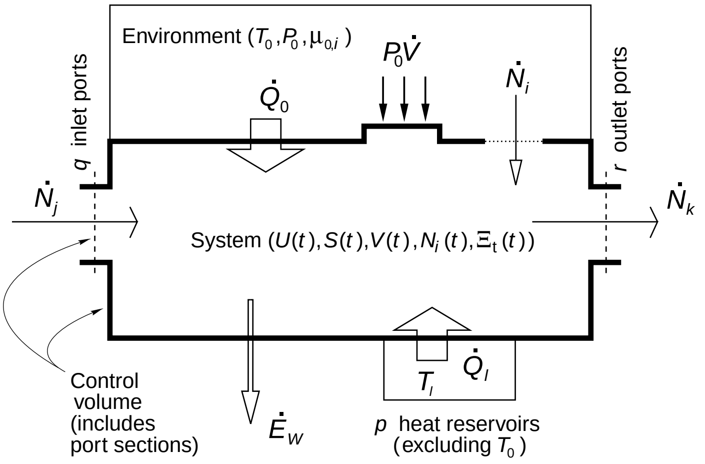

Environment (T_0, P_0, $\mu_{0,i}$)
$P_0\dot{V}$
$\dot{N}_i$
$\dot{Q}_0$
q inlet ports
r outlet ports
$\dot{N}_j$
$\dot{N}_k$
System ($U(t)$, $S(t)$, $V(t)$, $N_i(t)$, $\Xi_t(t)$)
Control volume (includes port sections)
$\dot{E}_W$
T_l
$\dot{Q}_l$
p heat reservoirs (excluding T_0)

Figure B.2 *A generalised non-steady-state steady-flow process (NSSF) based on total exergy but ignoring kinetic and gravitational potential exergies in the process streams.*

B.3.1 Total nonflow exergy

The nonflow total exergy inventory Ξ_t of the process is defined by:

$$\Xi_\mathrm{t} = U + P_0 V - T_0 S - \sum_{i=1}^{n} \mu_{0,i}\, N_i \; + \; \begin{array}{l}\text{other forms of macroscopic}\\ \text{exergy inventory}\end{array} \tag{B.30}$$

where: $_0$ indicates the dead-state value

B.3.2 Total flow exergy (excluding kinetic and GP exergy)

The molal total flow exergy $\bar{e}_\mathrm{t}$ at each inlet port ($j = 1 \ldots q$) and outlet port ($k = 1 \ldots r$) cross-section is defined by:

$$\bar{e}_\mathrm{t} = \bar{h} - T_0 \bar{s} - \sum_{i=1}^{n} \mu_{0,i}\, \frac{\dot{N}_i}{\dot{N}}$$

where: $\bar{h} = \bar{u} + P\bar{v}$ is the molal enthalpy

Introducing x_i, the mole fraction of the i^{th} component at port j (or k):

$$x_i = \frac{\dot{N}_i}{\dot{N}}, \qquad \text{with:} \;\; \dot{N} = \sum_{i=1}^{n} \dot{N}_i \tag{B.31}$$

Hence:

$$\bar{e}_\mathrm{t} = \bar{h} - T_0 \bar{s} - \sum_{i=1}^{n} \mu_{0,i}\, x_i \tag{B.32}$$

B.3.3 Non-stream heat transfer exergy

The exergy associated with heat transfer from $\ell = 1 \ldots p$ external heat reservoirs is given by (as before):

$$(\dot{E}_Q)_\ell = \left(1 - \frac{T_0}{T_\ell}\right) \dot{Q}_\ell \tag{B.22}$$

B.3.4 Full dead-state definition

A full dead-state definition requires the following information in appropriate form:

$$T_0,\ P_0,\ \mu_{0,i} \qquad \text{or} \qquad T_0,\ P_0,\ \bar{h}_{0,i},\ \bar{s}_{0,i}$$

B.3.5 Mass, energy, and entropy balances

The $i = 1 \ldots n$ mass balances are given by:[48]

$$\frac{dN_i}{dt} = \sum_{j=1}^{q} (\dot{N}_i)_j - \sum_{k=1}^{r} (\dot{N}_i)_k + \dot{N}_i \qquad i = 1 \ldots n \tag{B.33}$$

noting that:

$$\dot{N} = \sum_{i=1}^{n} \dot{N}_i \tag{B.34}$$

The energy balance is given by:

$$\frac{dU}{dt} = \sum_{\ell=0}^{p} \dot{Q}_\ell - \dot{W} + \sum_{j=1}^{q} (\dot{N}\bar{h})_j - \sum_{k=1}^{r} (\dot{N}\bar{h})_k + \sum_{i=1}^{n} \dot{N}_i \bar{h}_{0,i} \tag{B.35}$$

where the following shorthand applies:

$$(\dot{N}\bar{h})_j \ \equiv \ \sum_{i=1}^{n} (\dot{N}_i \bar{h}_i)_j \quad etc \tag{B.36}$$

The entropy balance is given by:

$$\dot{S}_{\text{gen}} = \frac{dS}{dt} - \sum_{\ell=0}^{p} \frac{\dot{Q}_\ell}{T_\ell} - \sum_{j=1}^{q} (\dot{N}\bar{s})_j + \sum_{k=1}^{r} (\dot{N}\bar{s})_k + \sum_{i=1}^{n} \dot{N}_i \bar{s}_{0,i} > 0 \tag{B.37}$$

where the following shorthand applies:

$$(\dot{N}\bar{s})_j \ \equiv \ \sum_{i=1}^{n} (\dot{N}_i \bar{s}_i)_j \quad etc \tag{B.38}$$

B.3.6 Chemical exergies for industrial fuels

Exergy analysis of processes involving combustion requires the chemical exergy of the fuel stream to be known.[49] Information on the assessment of chemical exergy

[48] $\dot{N}_i$ represents the *net* exchange of particles.

[49] This viewpoint assumes the fuel is already at a restricted dead-state, *ie*, at temperature and pressure equilibrium with its environment, and without velocity or elevation.

can be found in: Bejan (1997, p 359), Kotas (1995, appendix C), Nieuwlaar (1996, p 171), Stepanov (1995).

B.3.7 Closure

Total exergy as given here represents complete exergy only insofar as the dead-state is unrestricted — the kinetic and GP terms introduced with thermomechanical exergy have been omitted. Hence, total exergy, as is given here, is *not* a generalisation of thermomechanical exergy. □

Appendix C

Code developed to assist porting

This appendix lists examples of code developed to assist the porting of deeco.

C.1 Introduction

No major blocks of new code were required as part of the *deeco* port. However a number of small C++ programs were written to test the behaviour of the SCO UnixWare 2.1.2 SDK / x86 development environment *vis-à-vis* the original Hewlett-Packard HP-UX 8.07 environment (see p 154). In addition, KornShell and PERL scripts were used to automate repetitive tasks. Three examples are given in this appendix:

- USL Standard Components graph support trials
- program listings comparision utility
- *deeco* output comparison utility

C.2 USL Standard Components graph support trials (C++ programming)

The `widget3` program was developed from tutorial code in Weitzen (1992). It provided important information concerning differences between the two platforms. C++ references based on a language implementation similar to that used by *deeco* include: Ellis and Stroustrup (1992), Lippman (1995), Stroustrup (1991).

The program listing for `widget3` is contained on p 200–206.

C.3 Program listings comparison utility (shell scripting)

A number of shell scripts were developed to assist with the *deeco* port. By way of example, the KornShell script `.miff` reports differences between two nominated versions of the code-base. The KornShell is the default shell on UW2. KornShell scripting is described by Rosenblatt (1993).

The program listing for `.miff` is given on p 207–208.

C.4 *deeco* output comparison utility (PERL scripting)

The PERL script `pov` was used to scan *deeco* output files from the two platforms and report numerical discrepencies for subsequent checking by hand. PERL scripting is described by Wall *et al* (1996).

The program listing for `pov` is given on p 209. □

widget3

make3

```
#   FILENAME:     make3
#   PROJECT:      Widget manufacturing tutorial using SC graph library
#   SYSTEM:       SCO UnixWare 2.1.2 / x86  (SVR4.2 based)
#   COMPILER:     SCO SDK CC  (Edison Design Group C++ front end)

#   ------------
#   Command Line
#   ------------

#   $ make [-f makefile_name] 2>&1 | tee -a make.log      # a is append

#   -----------------
#   Macro Definitions
#   -----------------

CC       =   CC            # system C++ compiler

LIBES    =  -lm -l++       # libm.a and lib++.a essential; order probably irrelevant
                           # lib++.a almost certainly contains whole USL SC suite
                           # anyhow libGraph.a and libGA.a are not present on this system

INCL     =  #

CCFLAGS  =  -Xo -g #-v     # v is verbose (remarks also reported)
                           # g is debug info
                           # Tlist changes template instantiation
                           # Xstr sets language dialect

#   ---------------------
#   Compiler Instructions
#   ---------------------

OBJECTS =       widget3.o
widget3 :       $(OBJECTS)
                $(CC) $(CCFLAGS) $(INCL) $(OBJECTS) $(LIBES) -o widget3

widget_HEADS =  DGraph3.h
widget3.o :     $(widget_HEADS) widget3.C
                $(CC) -c $(CCFLAGS) $(INCL) widget3.C

#   FILE ENDS
```

widget3.C

```
/////////////////////////////////////////////////////////////////
//
// WIDGET PROGRAM : widget3
//
/////////////////////////////////////////////////////////////////
//
// Robbie Morrison
// widget3.C
//
// Widget manufacturing example
//
/////////////////////////////////////////////////////////////////
```

```
/////////////////////
// Revision log
/////////////////////

// 15.07.98 - files moved to SUZIE
// 15.07.98 - compile obtained / runs to completion / output not correct
// 16.07.98 - 'define widget' sequence corrected
// 16.07.98 - DGraph, DVertex, DEdge notation added and style matched to 'deeco' code
// 16.07.98 - DGraph.* files peeled off, but .C code stripped into .h due to linker error
// 23.10.98 - -Xo compiler switch added to make3 / code appears to function okay

//////////////
// Notes
//////////////

// Based on:  Weitzen, TC.  1992.  The C++ graph classes : a tutorial.
// In - AT&T and UNIX Systems Laboratories (eds).  1992.  USL C++ Standard
// Components, Release 3.0 Programmers Guide.  [Hewlett-Packard Handbook]

// User-defined      Base class     Notes
// ------------------------------------------
// DGraph            Graph          directed graph
// DVertex           Vertex         contains data member for identification text
// DEdge             Edge           contains data member for edge weighting
                                    (weighting is a non-negative integer)

// Container class    Library       Notes
// -------------------------------------------
// Set_of_p<T>        Set           optimised version of Set<T*>
// Set_of_piter       Set           Set_of_p iterator
// List_of_p<T>       List          optimised version of List<T*>
// List_of_piter      List          List_of_p iterator

//////////////////////
// Headers
//////////////////////

#include <iostream.h>
#include <stdlib.h>                     // for exit()
#include "DGraph3.h"                    // check revision is correct

//////////////////////////
// Utility functions
//////////////////////////

// List to Set conversion function

Set_of_p<DVertex>
listp_to_setp(List_of_p<DVertex> lp)
{
  Set_of_p<DVertex> sm;                 // declare DVertex pointer Set to hold solution set
  List_of_piter<DVertex> lpi (lp);      // local templatised List class constructor call
  DVertex* mp;                          // .. taking a parameter passsed from the function
                                        // .. as an argument
  while (lpi.next(mp)) {                // next() is List class member function
    sm.insert(mp);                      // insert() is Set class member function
  }
  return sm;
}
```

```
// Print function

void
print(Set_of_p<DVertex> mset)
{
  Set_of_piter<DVertex> sm (mset);      // local templatised Set class constructor call
  DVertex* mp;                          // .. taking a parameter passsed from the function
                                        // .. as an argument
  while (mp = sm.next()) {              // next() is Set class member function; assigment
                                        // ..  is correct
    cout << mp->id                      // ->id is of type String
         << " "
         << flush;                      // member function: ostream &ostream::flush(); causes
  }                                     // .. the ostream object's buffers to be flushed,
  cout << "\n";                         // .. forcing the contents to be written to the
}                                       // .. actual device connected to the ostream object

////////////////////////////
// Predicate functions                  // a predicate is a function used to control an
////////////////////////////            // .. algorithm; in this case a return of 0
                                        // .. terminates the algorithm; predicates are
                                        // .. typically called using function pointers
                                        // .. [SS 3/e p63]; note a pointer to a function
                                        // .. may be used without an explicit '*' (revised
                                        // .. under ANSI C) [Lewine 1991 p571]

// Global declarations                  // these invoke constructors prior to main()
                                        // note 'static' is the default for global variables

static DGraph prod;                     // used to transfer argument to predicate functions
                                        // .. (poor practice)
static List_of_p<DVertex> dfunc_list;   // templatised List class nil-argument constructor
                                        // .. call; dfunc_list is used by dfunc()

// Predicate dfunc                      // required for example 4
                                        // never returns 0 to terminate algorithm, used for
                                        // .. side effects
int
dfunc(DVertex* m)
{
  Set_of_piter<DEdge> si (m->out_edges_g(prod));  // templatised Set class contructor call,
                                                  // .. out_edges_g() is a Vertex class
                                                  // .. member which returns Set_of_p<Edge>
                                                  // .. noting that DEdge 'is an' Edge

  DEdge* t;
  while (t = si.next()) {               // next() is Set class member function; 'while'
                                        // .. block is based on next() return value;
                                        // .. assignment is correct
    if (!t->ttime) {                    // if t->ttime is 0, step into block, as a DVertex
                                        // ..  with 0 transport time has been found
      dfunc_list.put(m);                // dfunct_list has global scope; put() is List class
                                        // .. member function
      break;                            // presume break is from 'while' block (it should be)
    }
  }
  return 1;                             // the tutorial comments: "don't terminate until out
}                                       // .. of Modules"

// Predicate d2func                     // required for example 3

int
d2func(DVertex* m)
{
  Set_of_piter<DEdge> si (m->out_edges_g(prod));
  DEdge* t;
  while (t = si.next()) {               // next() is Set class member function; assignment
                                        // .. is correct
    if (!t->ttime) {                    // if t->ttime is 0, return 0 to terminate algorithm
      return 0;
    }
  }
  return 1;
}
```

```
///////////////////////
// Main commences
///////////////////////

int
main(int argc, char* argv[])            //command line arguments
{

//////////////////////
// Define widget
//////////////////////

// Refer: Fig 1 in tutorial for diagramatic representation of 'widget'.

// Note: <DGraph.h> contains #define dv(s) (DVertex::m[s]) which
// provides a pointer to the identified DVertex if this vertex has
// previously been instanciated; insert() is a Graph class member function.

// Note: The following code dynamically declares DVertices and DEdges
// without requiring the direct use of object names.

DGraph widget;

widget.insert(new DEdge(new DVertex("A"), new DVertex("D"), 5));
widget.insert(new DEdge(         dv("A"), new DVertex("F"), 1));
widget.insert(new DEdge(         dv("D"), new DVertex("C"), 2));
widget.insert(new DEdge(         dv("D"), new DVertex("J"), 0));
widget.insert(new DEdge(new DVertex("M"),          dv("J"), 0));
widget.insert(new DEdge(         dv("C"), new DVertex("B"), 0));
widget.insert(new DEdge(         dv("J"), new DVertex("K"), 6));
widget.insert(new DEdge(         dv("J"),          dv("B"), 7));
widget.insert(new DEdge(         dv("K"), new DVertex("Z"), 8));
widget.insert(new DEdge(         dv("B"),          dv("Z"), 9));

//////////////////////
// Functionality
//////////////////////

// Note: dfs() is a Graph class member function. It is a depth-first-search
// graph traversal algorithm.  The declaration for dfs(), using derived
// classes, is:
//
//   typedef int GApredicate(DVertex)(const DVertex* v);
//   List_of_p<DVertex> dfs(DGraph& g, DVertex*, GApredicate(DVertex)* f = 0);
//
// The client-defined function 'f' is optional, but should return type 'int',
// and 0 to terminate or 1 to continue.  dfs() returns a list of vertex pointers in
// order of visitation.

// Example 1

cout << "Example 1 - Where is module K needed? [Z]\n"
     << flush;

List_of_p<DVertex> mlist = dfs(widget, dv("K"));  // dfs() discussed above
Set_of_p<DVertex> m1set = listp_to_setp(mlist);   // user-defined function
print(m1set);                                      // user-defined function

prod = widget;                          // overloaded Graph class assignment operator
                                        // this passes an argument to dfunc() or d2func()
                                        // .. (poor practice)
                                        // prod has global scope
// Example 2

cout << "Example 2 - Which modules are immediately needed to compose module K? [J]\n"
     << flush;
Set_of_p<DVertex> mset;                 // declare DVertex pointer Set to hold solution set
Set_of_piter<DEdge> si (dv("K")->in_edges_g(widget));  // constructor call
DEdge* t;
while (t = si.next()) {                 // next() is Setiter class member function
  mset.insert(t->src());                // src() is Edge class member function
}
print(mset);                            // user-defined function

// Example 3                            // refer to tutorial, section 5.1

cout << "Example 3 - Which is the first 0 edged vertex found? [BCDJM / ending D]\n"
     << flush;

List_of_p<DVertex> m6list = dfs(widget, dv("A"), d2func);  // dfs() discussed above
cout << m6list.tail()->id               // tail() is an List class queue operation member
                                        // .. function
     << "\n"                            // ->id is of type String
     << flush;

// Example 4

cout << "Example 4 - Which are the 0 edged vertices? [BCDJM / DC]\n"
     << flush;

List_of_p<DVertex> m5list = dfs(widget, dv("A"), dfunc);  // dfs() discussed above
List_of_piter<DVertex> lim5 (dfunc_list);         // contructor call; argument dfunc_list
DVertex* mp;                                      // .. has global scope; List_of_p<DVertex>
                                                  // .. already constructed
while (lim5.next(mp)) {                           // next() is List_of_piter class member
                                                  // .. function
  cout << mp->id                                  // ->id is of type String
       << " "
       << flush;
}
cout << "\n";

// Example 5 - additional

cout << "Example 5 - List of vertices. [ABCDFJKMZ]\n";

print(listp_to_setp(m5list));                     // user-defined function

/////////////////
// Clean up
/////////////////

Set_of_piter<DVertex> si2 (widget.vertices());    // identify vertices
DVertex* mp2;
while (mp2 = si2.next()) {                        // cycle thru vertices
  Set_of_piter<DEdge> si3 (mp2->edges());         // identify edges
  DEdge* t;
  while (t = si3.next()) {                        // cycle thru edges
    delete t;                                     // return memory used by edge
  }
  delete mp2;                                     // return memory used by vertex
}

//////////////////
// Main ends
//////////////////

exit(0);                                          // return value for main(), 0 is success
}

// FILE ENDS
```

DGraph3.h

```
/////////////////////////////////////////////////////////////////
//
// GRAPH MANAGEMENT OBJECT
//
/////////////////////////////////////////////////////////////////
//
// Robbie Morrison
// DGraph3.h
//
/////////////////////////////////////////////////////////////////

#ifndef DGRAPH_H_INCLUDED
#define DGRAPH_H_INCLUDED

//////////////
// Notes
//////////////

// USL Standard Components use macro invocations to simulate
// genericity, rather than templates.  Hence the rather unusual
// code.  Refer to online documentation for more information.
// Note: no internal whitespace, no trailing semicolon, indent is okay.

// No DGraph.C file exists.  The final macro call:
// Graph_algimplement(...) was in a separate DGraph.C, but this caused
// a linker (ld) error: symbol 'm__7DVertex' defined twice.  This macro
// call was duly moved to the final call in this file.

////////////////
// Headers
////////////////

// SCO C++ Standard Components 3.1 headers
// -l++ compiler option mandatory

#include <Graph.h>           // includes <Set.h>
#include <Graph_alg.h>       // includes <Graph.h>, <List.h>, <Set.h>
#include <Map.h>             // Map is an associative array
#include <String.h>

#include <iostream.h>

////////////////////////////////////////////
// Preamble and first macro invocation
////////////////////////////////////////////

class DVertex;                         // forward declaration

typedef DVertex* DVptr;                // makes 'DVptr' a type in its own right
#define dv(s) (DVertex::m[s])          // macro function for convenience
                                       // 's' is key of type String

Graphdeclare1(DGraph,DVertex,DEdge)    // macro invocation in <Graph.h>

/////////////////////////////////////////////////////////////////
//
// CLASS : DGraph
//
/////////////////////////////////////////////////////////////////

class DGraph: public Graph {
  public:
    DGraph () : Graph () {                          // nil-argument constructor
    }
    derivedGraph(DGraph,DVertex,DEdge)              // macro invocation in <Graph.h>
};
```

```
/////////////////////////////////////////////////////////////////
//
// CLASS : DVertex
//
/////////////////////////////////////////////////////////////////

class DVertex: public Vertex {
  public:
    static Map<String, DVptr> m;                 // templatised associative array as class
                                                 // .. member, required because DVertices
                                                 // .. are dynamically instanced without
                                                 // .. an object name
    String id;                                   // holds DVertex object identification
                                                 // .. text; public (poor practice)
    DVertex (String s) : Vertex (), id (s) {
      m[s] = this;                               // 'this' is pointer to current object, aka
                                                 // .. self-reference

    }
    derivedVertex(DGraph,DVertex,DEdge)          // macro invocation in <Graph.h>
};

// Global declaration of m

Map<String, DVptr> DVertex::m;                   // second declaration for global scope; no
                                                 // DVertex objects constructed thus far

/////////////////////////////////////////////////////////////////
//
// CLASS : DEdge
//
/////////////////////////////////////////////////////////////////

class DEdge: public Edge {
  public:
    int ttime;                                   // weighting of edge; public (poor practice)

    DEdge (DVertex* m1, DVertex* m2, int time) : Edge (m1, m2), ttime (time) {
    }
    derivedEdge(DGraph,DVertex,DEdge)            // macro invocation in <Graph.h>
};

////////////////////////////////
// Final macro invocations
////////////////////////////////

Graphdeclare2(DGraph,DVertex,DEdge)              // macro invocation in <Graph.h>
Graph_algdeclare(DGraph,DVertex,DEdge)           // macro invocation in <Graph_alg.h>

// Was in DGraph.C preceded by #include "DGraph.h"
// but this caused linker error, as noted earlier

Graph_algimplement(DGraph,DVertex,DEdge)         // macro invocation in <Graph_alg.h>

#endif                                           // DGRAPH_H_INCLUDED

// FILE ENDS
```

.miff

```
#!/bin/ksh

#  Shell:     Korn
#  Name:      .miff (executable)
#  Author:    Robbie Morrison
#  Date:      1998
#  Status:    appears stable / runs in either main-shell or subshell mode
#  Purpose:   reports sdiff's for the deeco source code suite
#  Usage:     . ./.miff dir1 dir2 > output
#
#  WARNING:  directories must be ~/dir, although .miff can be elsewhere
#  Note:     sdiff exit states: 0 = identical / 1 = different / 2 = file missing

ver="01"                                # to keep track of modifications

print "++ Script:     $0"
print "   Date:       \c"; date         # \c omits trailing newline
print "   Version:    $ver"
print "   Arg count:  $#"
print "   Args:       $@"
print "   Note:       No response indicates both files are present and identical\n"

if [[ $# != 2 ]]; then                  # check number of arguments is two
    print "MISTAKE: Argument count not two as required"
    return 1                            # ends script in both main-shell and subshell mode
fi

DIR1=$1                                 # to improve code readability
DIR2=$2

print "Directory 1:  ~/$DIR1"           # ~ does not need escaping
print "Directory 2:  ~/$DIR2\n"

# COMPARE runs sdiff -s to identify differences
# arguments: dir1 dir2 file / output: sdiff -s output / returns: 1 if successful
function compare {

    echo  "---------------------"
    print "File: $3"
    echo  "---------------------"

    flag="clear"                        # reset flag each time
    if [[ ! -a ~/$1/$3 ]]; then         # check files are present
        print "ABSENT:  ~/$1/$3"
        flag="fail"
    fi
    if [[ ! -a ~/$2/$3 ]]; then
        print "ABSENT:  ~/$2/$3"
        flag="fail"
    fi
    if [[ $flag != clear ]]; then
        print
        return 0                        # quit function at this point
    fi

    sdiff -s -w180 ~/$1/$3 ~/$2/$3      # -s report only different lines / -w set cols
    exs=$?                              # $? is exit status from previous command
    if [[ $exs = 1 ]]; then             # 1 indicates differences / only one = is
                                          required in ksh
        print
    fi
    return 1
}
```

```
for file in                         \
                                    \
    ’App.h’      ’App.C’            \
    ’Balan.h’    ’Balan.C’          \
    ’Collect.h’  ’Collect.C’        \
    ’Connect.h’  ’Connect.C’        \
    ’Convers.h’  ’Convers.C’        \
    ’Data.h’     ’Data.C’           \
    ’deeco.C’                       \
    ’deecoApp.h’ ’deecoApp.C’       \
    ’Demand.h’   ’Demand.C’         \
    ’DGraph.h’   ’DGraph.C’         \
    ’Net.h’      ’Net.C’            \
    ’Network.h’  ’Network.C’        \
    ’Port.h’     ’Port.C’           \
    ’Proc.h’     ’Proc.C’           \
    ’ProcType.h’ ’ProcType.C’       \
    ’Scen.h’     ’Scen.C’           \
    ’Simplex.h’  ’Simplex.C’        \
    ’Storage.h’  ’Storage.C’        \
    ’TestFlag.h’                    \
    ’nr.h’                          \
    ’nrutil.h’   ’nrutil.c’         \
    ’simp1.c’ ’simp2.c’ ’simp3.c’   \
    ’xsimplx.c’                     \
    ’deeco.msg’                     \
    ’except.h’                      \
    ’makefile’                      \
    ’robbie’; do                        # ’except.h’ was added by Richard O’Keefe
                                        # ’makefile’ was from deeco 0.5 suite
                                        # ’robbie’ is my makefile

    compare $DIR1 $DIR2 $file           # function call
done

print "\n++ Script complete\n"

# ===
```

pov

```
#! /usr/bin/perl -c
# -dD14

# switches: -c check syntax -dD14 report stuff
# usage: pov <newfile> <oldfile> <output>

$hats = 8;                    # number of hats

@ARGV != 3 || die "Argument count is @ARGV and not 3\n";

$newfile = $ARGV[0];
$oldfile = $ARGV[1];
$outfile = $ARGV[2];

open(NEW, $newfile) || die "Cannot open $newfile: $!\n";
open(OLD, $oldfile) || die "Cannot open $oldfile: $!\n";
open(OUT, $outfile) || die "Cannot open $outfile: $!\n";

while(<NEW>) {
#   next if /^$/;             # ignore blank lines / not sure of this
    next if /^%/;             # ignore 'deeco' comment lines
    s/\s+$//g;                # strip trailing spaces if any
    s/\s+/\t/g;               # replace spaces with tabs
    s/?nan/\?   /g;           # replace '?nan' with '?   '
    s/?inf/\?   /g;           # replace '?inf' with '?   '
    push(@new, $_);           # build '@new'
}
close (NEW);

while(<OLD>) {
    next if /^$/;             # ignore blank lines
    next if /^%/;             # ignore 'deeco' comment lines
    s/\s+$//g;                # strip trailing spaces if any
    s/\s+/\t/g;               # replace spaces with tabs
    s/?nan/\?   /g;           # replace '?nan' with '?   '
    s/?inf/\?   /g;           # replace '?inf' with '?   '
    push(@old, $_);           # build '@old'
}
close (OLD);

@newsort = sort @new;         # sort array by record
@oldsort = sort @old;         # sort array by record

while (1) {
    $newbuf = pop(@newsort);
    $oldbut = pop(@oldsort);
    if $newbuf && $oldbuf {
        last;
    }
    if $newbuf ne $oldbuf {
        $underline = "\t\t";
        for ($n = 2, $n <= 5, $n++) {
            if split(/,/,$newbuf)[$i] ne split(/,/,$oldbuf)[$i] {
                $underline .= "^" x $hats;
                $underline .= "\t";
            }
            else
                $underline .= "\t";
            }
        }
        $underline .= "\n";
    }
    print OUT "<", $newbuf, ">", $oldbuf, $underline, "\n";
}

close (OUT);

# ===
```

Appendix D

Code for MCFP algorithm

This appendix contains a listing for the out-of-kilter algorithm.

D.1 Introduction

A number of graph theory and flow network algorithms were translated into MATLAB 5.3 / Linux from BASIC, Pascal, and pseudo-code listings found within the literature. The purpose of this exercise was to learn about the methods which underpin *deeco*.

D.2 Out-of-kilter algorithm (MATLAB scripting)

The program ook13 was translated into MATLAB 5.3 from a Pascal listing in Smith (1982, p 125–129). Further details can be found in the comments contained within the listing. It should be noted that *deeco* does not use a flow network optimisation algorithm to determine a minimum cost routing, but instead the task is restated in terms of matrix-based linear algebra and solved using the simplex method. Mathematically, these two formulations — based on graph theory and linear programming respectively — can be shown to be equivalent, as discussed on p 90.

The program listing for ook13 is given on p 211–218. □

ook13.m

```
function ook13
%OOK13    Least-cost network flow algorithm -- version 13
%   OOK prompts for an external data file and then determines the least-cost
%   flow in the specified network using Ford and Fulkerson's 'out-of-kilter'
%   algorithm.  Data-file is tab-delimited ASCII thus (\t = tab, \n = newline):
%
%     startNode\tfinishNode\tlowerBound\tupperBound\tunitFlowCost\n
%
%   Nodes must be numbered consecutively from 1 without gaps.
%   See listing for more information.  To do so, enter TYPE OOK13.
%
%   Adapted from: Smith, David K. 1982. "Network optimisation practice :
%   a computational guide". Ellis Horwood, Chichester. ISBN 0-85312-403-5.
%   Pages 111-129.  Formal algorithm on pages 121-122.  Worked example 4.4.1.
%   Originally coded in Pascal and line-numbered BASIC.
%
%   Smith also lists a number of adaptations which allow the algorithm to be
%   applied to lesser problems, including 'maximum flow' and 'shortest path'.
%   Smith also explains how to formulate least-cost network flow as a Simplex
%   optimisation problem.
%
%   In addition, see: Ford, L.R. and D.R. Fulkerson. 1962. "Flows in networks".
%   Princeton University Press.

%   PROGRAM NOTES
%
%   Written:  Robbie Morrison
%   Date:     14 April 1999
%   Status:   tested on Smith's worked example 4.4.1
%             spurious reporting of edges from INKILTER print and plot statements
%             (of no real significance)
%   Language: MATLAB 5.3 / Linux
%   Note:     a number of small modifications would be needed to port to MATLAB Student 4.0
%
%   This code is based on the Pascal listing in Smith on pages 125-129.
%   The graph theory terminology used here is: nodes/edges.
%   Nodes must be numbered consecutively from 1. Otherwise an error is reported.
%
%   MISCELLENEOUS NOTES
%
%   Data-entry is via console in Smith, whereas here data-entry is by ASCII data-file.
%   Constants 'maxnodes' and 'maxarcs' not needed as arrays are dynamic in MATLAB.
%   'INF' set to IEEE floating-point 'Inf' if supported (ie, Linux, Windows, Mac).
%   Pascal 'REPEAT-code-UNTIL-this' replaced by 'WHILE 1-code-IF-this, BREAK, END, END'.
%   Pascal 'pass-by-reference' function argument achieved using GLOBAL variable declarations.
%   No explicit boolean data-type in MATLAB, mimiced by using 'TRUE' = 1 and 'FALSE' = 0.
%   Variables begining with an upper-case character are GLOBAL by convention.
%   Variables treated as constants are written in capitals (no actual contants in MATLAB).
%   All vectors are defined as row vectors.
%   Smith (1982) is held by the National Library of New Zealand.


disp(' ')
disp('Out-of-kilter algorithm for finding the')
disp('least-cost flow in a given network.')

% ------------------
% preliminaries
% ------------------

clear variables              % clears workspace, not particularly neighbourly
format compact               % set console format as compact
more off                     % allow screen output to run uninterupted
```

```
% -------------------------
% set globals and constants
% -------------------------

% variables declared GLOBAL

global Plus Minus             % simulate 'pass-by-reference' in subfunctions, initialised null
global TRUE FALSE             % make accessible to subfunctions
global edge1 i j

% variables treated as constants (constants are not enforced by the language)

TRUE  = 1;                    % 'TRUE' and 'FALSE' used as boolean values
FALSE = 0;

% set infinity

if isieee                     % test for IEEE floating-point arithmetic
   INF = Inf;                 % use IEEE 'Inf'
else
   INF = 9999;                % alternative strategy
end

% ------------------
% key variables
% ------------------

% variable   Smith        description
% ..........................................................
%
% filename    -           name of data-file (entered by user)
% fid         -           file descriptor (returned by FOPEN call)
%
% data        -           matrix generated from data-file
%
% nodes       nodes       total number of nodes in network
% edges       arcs        total number of edges in network
%
% i           i           vector of start nodes for each edge
% j           j           vector of finish nodes for each edge
% l           l           vector of lower bounds for each edge
% u           u           vector of upper bounds for each edge
% c           c           vector of unit-flow costs for each edge
%
% x           x           vector of the current flow in each edge
% pi          pi          vector of pi-multipliers for each node
% a           a           vector of first part of two part node labels
%                            abs(a(node)):  preceding node in path from 's'
%                            sign(a(node)): forward edge = +, reverse = -
%                            a(node) = 0:   node not labelled
% b           b           vector of second part of two part node labels
%                            b(node):       potential flow augmentation
%
% node        node        node counter
% edge        arc         primary edge counter and index of current out-of-kilter edge
% edge1       arc1        secondary edge counter
%
% Plus        plus        permitted increase in flow for current out-of-kilter edge
% Minus       minus       permitted decrease in flow for current out-of-kilter edge
%
% pm          pm          indicator for whether an edge in the flow-augmenting path is
%                         .. forward or reverse
% change      change      flow increase/decrease
% delta       del         change in the pi-multipliers for the unlabelled nodes
% newlabels   newlabels   counter for the number of new labels assigned in a pass thru
%                         .. the edges
%
% s           s           start of the out-of-kilter edge
% t           t           finish of the out-of-kilter edge
%
```

```
% infeasible  infeasible  boolean indicates whether value of 'delta' can be assigned or not
% accept      accept      boolean showing flow exists within limits
% ok          ok          boolean variable used as dummy
%
% k           -           integer counter
%
% time        -           elapsed time
% cpu         -           number of floating point operations found using FLOPS

% ------------------
% subfunctions
% ------------------

% name          arguments    return        description
% ..........................................................
%
% NODUPLICATE   row vector   row vector    sorts row vector and removes duplicates
% INKILTER      see code     boolean       checks edge, updates 'Plus' and 'Minus'

% ------------------
% read in dataset
% ------------------

% open data-file

disp(' ')
fid = 0;
while fid < 1
   filename = input('Open data-file (tab-delimited): ','s');
   [fid,message] = fopen(filename,'r');           % open as read only
   if fid == -1
      disp(message)
   end
end

% obtain data using DLMREAD

dlm  = '\t';                                       % set ASCII delimiter to [tab]
data = dlmread(filename, dlm);

% ------------------
% condition data
% ------------------

% make individual vectors

i = data(:,1)';
j = data(:,2)';
l = data(:,3)';
u = data(:,4)';
c = data(:,5)';

% check integrity

if ( all(0 <= l) == FALSE ) | ( all(l <= u) == FALSE )
   warning('OOK: Corrupt data, 0 <= l <= u false for at least one edge')
end

% determine total number of edges and nodes

nodevector = [data(:,1)' data(:,2)'];
nodelist  = noduplicate(nodevector);    % NODUPLICATE is a subfunction
nodes = length(nodelist);               % total number of nodes

edges  = size(data,1);                  % total number of edges

% check integrity

if nodes ~= nodelist(end)
   warning('OOK: Corrupt data, check node numbering for gaps')
end
```

```
% ------------------
% report statistics
% ------------------

fprintf(1,'\n')
fprintf(1,'  Data file:           ''%s''\n',filename)
fprintf(1,'  Number of nodes:      %d\n',nodes)
fprintf(1,'  Number of edges:      %d\n',edges)
fprintf(1,'  List of nodes:        ')
for k = 1:nodes
   fprintf(1,'%d ',nodelist(k))
end
fprintf(1,'\n')
fprintf(1,'  Infinity in use:      %d\n',INF)

% -------------------------------------------
% step 0 - initialise flows and pi-multipliers
% -------------------------------------------

tic                           % start timer
flops(0)                      % reset flop counter

infeasible = FALSE;           % assume feasible

% initialize flows and pi-multipliers

x  = zeros(1,edges);          % assign arbitrary flow to each edge subject to flow
                              % ..  conservation, in this case choose zero
pi = zeros(1,nodes);          % assign arbitrary 'pi' multiplier to each node, in this
                              % .. case choose zero

% ----------------------------------------
% step 0.5 - search for out-of-kilter edges
% ----------------------------------------

for edge = 1:edges
   while ( inkilter(x(edge), l(edge), u(edge), c(edge) +  pi(i(edge)) - pi(j(edge))) == FALSE ) ...
       & ( infeasible == FALSE )

% ----------------------------------
% step 1 - out-of-kilter edge located
% ----------------------------------

      if Plus > 0             % left of kilter-line, hence edge (t,s)
         s    = j(edge);
         t    = i(edge);
         a(s) = edge;
         b(s) = Plus;
      else                    % right of kilter-line, hence edge (s,t)
         s    = i(edge);
         t    = j(edge);
         a(s) = -edge;
         b(s) = Minus;
      end

      while 1                 % note BREAK statement marked Z which exits this loop

% ------------------------------------------------
% step 2 - find flow-augmenting path from 's' to 't'
% ------------------------------------------------

         % reset node labels excepting the labels for 's'

         for node = 1:nodes
            if node ~= s
               a(node) = 0;
               b(node) = 0;
            end
         end
```

```
% double loop (WHILE / FOR) thru edges, as use of an edge may mean an earlier
% .. unusable edge can now be employed

while 1
   newlabels = 0;                        % used as a boolean
   for edge1 = 1:edges
      if ( ( a(i(edge1)) == 0 ) & ( a(j(edge1)) ~= 0 ) ) | ( ( a(i(edge1)) ~= 0 ) ...
       & ( a(j(edge1)) == 0 ) )
         % above line indentifies edge with one end labelled and the other unlabelled
         ok = inkilter(x(edge1), l(edge1), u(edge1), c(edge1) + pi(i(edge1)) - pi(j(edge1)));
         % above line used for side-effects of 'Plus' and 'Minus', 'ok' is not used
         if ( a(i(edge1)) ~= 0 ) & ( Plus > 0 )
            % forward edge as indicated by first condition
            newlabels   = newlabels + 1;
            a(j(edge1)) = edge1;
            b(j(edge1)) = min(b(i(edge1)), Plus);
         elseif ( a(j(edge1)) ~= 0 ) & ( Minus > 0 )   % needed when 'a(j)' has just
                                                       % .. been set [original comment]
            % reverse edge as indicated be first condition
            newlabels   = newlabels + 1;
            a(i(edge1)) = -edge1;
            b(i(edge1)) = min(b(j(edge1)), Minus);
         end
      end
   end
   if ( newlabels == 0 ) | ( a(t) ~= 0 )           % point A
      break
   end                                             % otherwise scan edges again
end

if a(t) ~= 0                             % point A to here if second condition
                                         % .. is true, ie, flow-augmenting path
                                         % ..exists

   % increase flow in the chain [original comment]

   node = t;
   change = b(t);

   % update node 's' first

   if a(s) > 0                                     % forward edge
      x(edge) = x(edge) + change;
   else                                            % reverse edge
      x(edge) = x(edge) - change;
   end

   % work back thru flow-augmenting path

   while 1
      edge1 = abs(a(node));
      if a(node) > 0                               % forward edge
         node = i(edge1);
         pm   = 1;
      else                                         % reverse edge
         node = j(edge1);
         pm   = -1;
      end
      x(edge1) = x(edge1) + pm*change;             % update flow
      if node == s                                 % 's' has been reached
         break                                     % update complete
      end
   end

else                          % point A to here if first condition only is
                              % .. true, ie, no flow-augmenting path exists
% ----------------------------------------------------------------------------------
% step 3 - otherwise examine edges, determine 'delta', and update 'pi-multipliers'
% ----------------------------------------------------------------------------------

            delta = INF;
            infeasible = TRUE;

            for edge1 = 1:edges
               if ( ( a(i(edge1)) == 0 ) & ( a(j(edge1)) ~= 0 ) ) | ( ( a(i(edge1)) ~= 0 ) ...
                & ( a(j(edge1)) == 0 ) )
                  % above line indentifies edge with one end labelled and the other unlabelled
                  accept = ( x(edge1) < u(edge1) ) & ( x(edge1) > l(edge1) );
                  % above line for convenience
                  if ( c(edge1) + pi(i(edge1)) - pi(j(edge1)) > 0 ) & ( ( accept == TRUE ) ...
                   | ( x(edge1) == l(edge1) ) )
                     delta = min(delta, c(edge1) + pi(i(edge1)) - pi(j(edge1)));
                     infeasible = FALSE;
                  end
                  if ( c(edge1) + pi(i(edge1)) - pi(j(edge1)) < 0 ) & ( ( accept == TRUE ) ...
                   | ( x(edge1) == u(edge1) ) )
                     delta = min(delta, -(c(edge1) + pi(i(edge1)) - pi(j(edge1))));
                     infeasible = FALSE;
                  end
               end
            end

            if delta == INF
               delta = abs(c(edge) + pi(i(edge)) - pi(j(edge)));
               if ( x(edge) >= l(edge) ) & ( x(edge) <= u(edge) )
                  infeasible = FALSE;
               end
            end

            if infeasible == TRUE
               warning('OOK: There is no feasible flow')
            else
               for node = 1:nodes
                  if a(node) == 0                 % the node is unlabelled
                     pi(node) = pi(node) + delta;
                  end
               end
            end

         end                                      % finish of point A block

         if ( inkilter(x(edge), l(edge), u(edge), c(edge) + pi(i(edge)) - pi(j(edge))) == TRUE ) ...
          | ( infeasible == TRUE )
            break                                 % point Z
         end
      end
   end
end

% algorithm complete

time = toc;                   % stop timer
cpu  = flops;                 % obtain flop count
```

```
%------------------
% report results
% ------------------

if infeasible == FALSE
   fprintf(1,'\n')
   fprintf(1,'\tEdge  \tStart \tFinish\tLower \tUpper \tCost  \tOptimal\n')
   fprintf(1,'\tnumber\tnode  \tnode  \tbound \tbound \t      \tflow\n')
   for edge = 1:edges
      fprintf(1,'\t%d    \t%d    \t%d    \t%d    \t%d    \t%d    \t%d\n', ...
              edge,i(edge),j(edge),l(edge),u(edge),c(edge),x(edge))
   end
   fprintf(1,'\n')
   fprintf(1,'\tNode  \tPi(n)\n')
   fprintf(1,'\tnumber\n')
   for node = 1:nodes
      fprintf(1,'\t%d    \t%d\n',node,pi(node))
    end
else
   warning('OOK: Cannot print results as ''infeasible'' set to true')
end

fprintf(1,'\n')
fprintf(1,'  Time in algorithm:    %fs\n',time)
fprintf(1,'  Number of flops:      %d\n',cpu)

for edge1 = 1:edge
   inkilter(x(edge1), l(edge1), u(edge1), c(edge1) + pi(i(edge1)) - pi(j(edge1)))
end

% ------------------
% housekeeping
% ------------------

clear k
more on

% script completes

% -------------------------
% subfunction - NODUPLICATE
% -------------------------

function out = noduplicate(in);
%NODUPLICATE returns sorted row vector with duplicate entries removed.

if ( size(in,1) ~= 1 ) | ( size(in,2) < 2 )
   warning('OOK: NODUPLICATE: Argument is not row vector')
end
in = sort(in);                % sort
out(1) = in(1);               % assign first element
for k=1:length(in)            % cycle thru elements of 'in'
   if in(k) ~= out(end)       % alternative: max(out)
      out = [out in(k)];      % append to 'out'
   end
end

% ----------------------
% subfunction - INKILTER
% ----------------------

function bool = inkilter(flow, low, high, pc)
%INKILTER tests edge to determine if it is out-of-kilter.  Also updates
%         values for 'Plus' and 'Minus'.

global TRUE FALSE             % to give access to these constants
global Plus Minus             % to simulate 'pass-by-reference'

global edge1 i j        % for INKILTER plot
```

```
bool  = FALSE;
Plus  = 0;                    % Smith used 'changeup'
Minus = 0;                    % Smith used 'changedown'

if nargin ~= 4
   warning('OOK: INKILTER: Argument count incorrect')
end

if pc > 0
   if flow < low
      Plus  = low - flow;
   elseif flow == low
      bool  = TRUE;
   elseif flow > low
      Minus = flow - low;
   end
elseif pc == 0
   if flow < low
      Plus  = high - flow;
   elseif ( flow >= low ) & ( flow <= high )
      Plus  = high - flow;
      Minus = flow - low;
      bool  = TRUE;
   elseif flow > high
      Minus = flow - low;
   end
elseif pc < 0
   if flow < high
      Plus  = high - flow;
   elseif flow == high
      bool  = TRUE;
   elseif flow > high
      Minus = flow - high;
   end
end

% plot kilter-line

clf              % clear existing figure window, otherwise open one
pause(0.1)       % to emphase plot changes

% set plot area (the following code is problem-dependent, unfortunately)

a = -1;          % x_min
b =  7;          % x_max
c = -5;          % y_min
d =  5;          % y_max
e =  0.5;        % edge_space

% determine lines

KL = [low d; low 0; high 0; high c]';          % kilter-line
XL = [a 0; b 0]';                              % y = 0
YL = [0 d; 0 c]';                              % x = 0

% plot information

axis([a-e b+e c-e d+e])
grid
hold on
plot(XL(1,:),XL(2,:),'Color','k')
plot(YL(1,:),YL(2,:),'Color','k')
plot(KL(1,:),KL(2,:),'LineWidth',2.0,'Color','r')
plot(flow,pc,'x','MarkerSize',18.0,'LineWidth',2.5,'Color','r')
title(['Edge (' num2str(i(edge1)) ', ' num2str(j(edge1)) ')'])
xlabel('edgeFlow')
ylabel('edgeCost - pi(beginNode) + pi(endNode)')
hold off

end

% ===
```

Appendix E

Notation

Note: only notation used in more than one place in the thesis is listed here.

Roman notation

A	: nonflow availability [J]
b	: mass-specific flow availability [J]
$\bar{E}$	: total energy (refer eqn (B.18)) [J]
$\bar{E}_0$	: dead-state total energy [J]
$\dot{E}$	: generalised flow exergy rate (requires subscripting) [W]
$\dot{E}_\mathrm{x}$	: thermomechanical flow exergy rate [W]
$\dot{E}_\mathrm{t}$	: total flow exergy rate [W]
$\dot{E}_Q$	: flow exergy rate associated with heat transfer [W]
$\dot{E}_W$	: flow exergy rate associated with work transer [W]
$\dot{E}_\mathrm{in}$	: flow exergy input rate [W]
$\dot{E}_\mathrm{out}$	: flow exergy output rate [W]
e_x	: mass-specific thermomechanical flow exergy [J/kg]
$\bar{e}_\mathrm{t}$	: molal total flow exergy [J/mol]
G	: Gibbs free energy [J]
g	: mass-specific Gibbs free energy [J/kg] : acceleration due to gravity [m/s^2]
$\bar{g}_{0,i}$	: partial molal dead-state Gibbs free energy for species i [J/mol]
H	: enthalpy [J]

h : mass-specific enthalpy [J/kg]

h_0 : mass-specific dead-state enthalpy [J/kg]

$\bar{h}$: molal enthalpy [J/mol]

$\bar{h}_{0,i}$: partial molal dead-state enthalpy for species i [J/mol]

h° : mass-specific methalpy (see footnote 43, p 192) [J/kg]

I : rotational inertia [kg m^2]

$\dot{L}$: generalised flow ϵnergy rate (requires subscripting) [W]

$\dot{L}_\mathrm{x}$: thermomechanical flow ϵnergy rate [W]

$\dot{L}_\mathrm{t}$: total flow ϵnergy rate [W]

$\dot{L}_Q$: flow ϵnergy rate associated with heat transfer [W]

$\dot{L}_W$: flow ϵnergy rate associated with work transfer [W]

ℓ_x : mass-specific thermomechanical flow ϵnergy [J/kg]

$\bar{\ell}_\mathrm{t}$: molal total flow ϵnergy [J/mol]

M : mass (notation as per Bejan (1997)) [kg]

$\dot{m}$: mass flow-rate [kg/s]

$\dot{m}_i$: mass flow-rate for species i [kg/s]

n : number of species present [–]

N : number of moles of all species [–]

N_i : number of moles of species i [–]

$\dot{N}$: molal flow-rate [mol/s]

$\dot{N}_i$: partial molal flow-rate for species i [mol/s]

P : absolute (static) pressure [N/m^2]

P_0 : dead-state (ambient) absolute (static) pressure [N/m^2]

Q : heat [J]

$\dot{Q}_\ell$: heat transfer-rate from reservoir ℓ [W]

$\dot{Q}_0$: heat transfer-rate from ambient [W]

S : entropy [J/K]

S_0 : dead-state entropy [J/K]

$\dot{S}_\mathrm{gen}$: entropy generation rate [W/K]

s : mass-specific entropy [J/kg K]

s_0 : mass-specific dead-state entropy [J/kg K]

$\bar{s}$: molal entropy [J/mol K]

$\bar{s}_0$: molal dead-state entropy [J/mol K]

$\bar{s}_{0,i}$	: partial molal dead-state entropy for species i [J/mol K]
t	: time [s]
T	: absolute temperature [K]
T_0	: dead-state (ambient) absolute temperature [K]
T_ℓ	: absolute temperature of reservoir ℓ [K]
U	: internal energy [J]
U_0	: dead-state internal energy [J]
V	: volume [m^3]
V_0	: dead-state volume (*ie*, control region datum) [m^3]
υ	: bulk stream velocity (see footnote 21, p 184) [m/s]
W	: work [J]
$\dot{W}$	: work transfer-rate [W]
$\dot{W}_{\text{lost}}$	: exergy destruction rate [W]
x_i	: mole fraction (see eqn (B.31)) [–]
Z	: generalised flow exergy quality factor (refer eqn (5.17)) [–]
Z_{x}	: flow exergy quality factor for thermomechanical exergy [–]
Z_{t}	: flow exergy quality factor for total exergy [–]
Z_Q	: flow exergy quality factor for heat flow [–]
Z_W	: flow exergy quality factor for work flow [–]
Z'	: generalised nonflow exergy quality factor (refer eqn (5.22)) [–]
Z'_{x}	: nonflow exergy quality factor for thermomechanical exergy [–]
Z'_{t}	: nonflow exergy quality factor for total exergy [–]
z	: bulk altitude (*ie*, height) [m]
z_0	: dead-state bulk altitude (*ie*, ground level) [m]

Greek notation

η_{I}	: average ϵnergetic efficiency (refer eqn (5.16)) [–]
η'_{I}	: marginal ϵnergetic efficiency (refer eqn (5.16)) [–]
η_{II}	: average exergetic efficiency (refer eqn (5.15)) [–]
η'_{II}	: marginal exergetic efficiency (refer eqn (5.15)) [–]

Λ : generalised nonflow ϵnergy (requires subscripting) [J]

Λ_{x} : thermomechanical nonflow ϵnergy [J]

Λ_{t} : total nonflow ϵnergy [J]

λ_{x} : mass-specific thermomechanical nonflow ϵnergy [J/kg]

$\bar{\lambda}_{\mathrm{t}}$: molal total nonflow ϵnergy [J/mol]

μ : chemical potential for single species system [J/mol]

μ_i : chemical potential for species i [J/mol]

$\mu_{0,i}$: dead-state chemical potential for species i [J/mol]

μ_i^* : restricted dead-state chemical potential for species i [J/mol]

Ξ : generalised nonflow exergy (requires subscripting) [J]

Ξ_{x} : thermomechanical nonflow exergy [J]

Ξ_{t} : total nonflow exergy [J]

ξ_{x} : mass-specific thermomechanical nonflow exergy [J/kg]

$\bar{\xi}_{\mathrm{t}}$: molal total nonflow exergy [J/mol]

ρ : mass density [kg/m$_3$]

Σ : \$nergy [multi-dimensional]

Σ^E : \$nergy accounted using flow exergy [J/quantity]

Σ^L : \$nergy accounted using flow ϵnergy [J/quantity]

$\Sigma^\$$: \$nergy factor price [\$/quantity]

Σ^A : \$nergy factor attributes [as appropriate]

ω : rotational velocity [rad/s]

Operators

đ : inexact differential (see footnote 25, p 184)

$\vee$: or operator

$\setminus$: set subtraction operator

$\otimes$: symmetric difference operator (see footnote 10, p 125)

$^\top$: transpose operator

References

Ackermann, Thomas, Karen Garner, and Alister Gardiner. 1999. Wind power generation in weak grids : economic optimisation and power quality simulation. *Renewable Energy*. **18**(2): 205–221.

Aho, A.V., J. Hopcroft, and J.D. Ullman. 1983. *Data structures and algorithms*. Reading, Massachusetts: Addison-Wesley.

Avondo-Bodino, Giuseppe. 1979. Graph theory and operations research. Chapter 8, pp. 223–253. In: *Applications of graph theory*. Editors: Wilson, R.J. and L.W. Beineke. London: Academic Press.

Aylward, Gordon H. and Tristan Findlay. 1994. *SI chemical data*. Third edition. Brisbane: John Wiley and Sons. ISBN 0-471-33554-1.

Ayres, Robert U. 1994. *Information, entropy and progress : a new evolutionary paradigm*. New York: American Institute of Physics Press. ISBN 0-88318-911-9.

Ayres, Robert U. 1996. Statistical measures of unsustainability. *Ecological Economics*. **16**(3): 239–255.

Ayres, Robert U., Leslie W. Ayres, and Katalin Martinas. 1998. Exergy, waste accounting, and life-cycle analysis. *Energy*. **23**(5): 355–363.

Baines, James Talbot. 1989. An integrated framework for interpreting sustainable development : ecological principles and institutional arrangements for the sustainable development of natural and physical resources. Unpublished report. Ministry for the Environment. Wellington, New Zealand. March. [Author affiliated with Centre for Resource Management, Lincoln University, New Zealand].

Bak, Per. 1996. *How nature works : the science of self-organized criticality*. New York: Copernicus.

Bannister, Paul. 1991. *An experimental and analytical assessment of a steam Rankine solar thermal system*. PhD thesis. Australian National University. Canberra, Australia.

Bejan, Adrian. 1982. *Entropy generation through heat and fluid flow*. New York: John Wiley and Sons. ISBN 0-471-09438-2.

Bejan, Adrian. 1997. *Advanced engineering thermodynamics*. Second edition. New York: John Wiley and Sons. ISBN 0-471-14880-6.

Bejan, Adrian and Eden Mamut (eds). 1999. *Thermodynamic optimization of complex energy systems*. Dordrecht: Kluwer Academic Publishers. ISBN 0-7923-5726-4 pbk.

Bejan, Adrian, George Tsatsaronis, and Michael Moran. 1996. *Thermal design and optimization*. New York: John Wiley and Sons. ISBN 0-471-58467-3.

Bhattacharyya, Subhes C. 1996. Applied general equilibrium models for energy studies : a survey. *Energy Economics*. **18**: 145–164.

Blackwell, William A. 1968. *Mathematical modeling of physical networks*. New York: Macmillan.

Bose, Ranjan K. and G. Anandalingam. 1996. Sustainable urban energy-environment management with multiple objectives. *Energy*. **21**(4): 308–318.

Bossel, Hartmut. 1994. *Modeling and simulation*. Wellesley, Massachusetts: A.K. Peters. ISBN 3-528-05419-0.

Brown, Lester R., Christopher Flavin, Hilary French, Janet Abramovitz, Seth Dunn, Gary Gardner, Ashley Mattoon, Anne Platt McGinn, Molly O'Meara, Michael Renner, David Roodman, Payal Sampat, John Tuxill, and Linda Starke. 1999. *State of the world 1999 : a Worldwatch Institute report on progress toward a sustainable society*. New York: Worldwatch Institute. ISBN 0-393-31815-X pbk.

Bruckner, Thomas. 1997. *Dynamische Energie- und Emissionsoptimierung regionaler Energiesysteme*. PhD thesis. Institut für Theoretische Physik, Universitat Würzburg. Germany. [In German].

Bruckner, Thomas, Helmuth-M. Groscurth, and Reiner Kümmel. 1997. Competition and synergy between energy technologies in municipal energy systems. *Energy*. **22**(10): 1005–1014.

Bunn, Derek W. and Erik R. Larsen (eds). 1997a. *Systems modelling for energy policy*. Chichester: John Wiley and Sons. ISBN 0-471-95794-1.

Bunn, Derek W. and Erik R. Larsen. 1997b. Systems modelling for energy policy. Chapter 1, pp. 1–8. In: *Systems modelling for energy policy*. Editors: Bunn, Derek W. and Erik R. Larsen. Chichester: John Wiley and Sons. ISBN 0-471-95794-1.

Busacker, R.G. and P.J. Gowen. 1961. *A procedure for determining a family of minimum-cost network flow patterns — ORO Technical Report 15*. Baltimore, Maryland: Operations Research Office, John Hopkins University.

Callen, Herbert B. 1985. *Thermodynamics and an introduction to thermostatistics*. Second edition. John Wiley and Sons. ISBN 0471862568 hbk.

Campbell, Colin J. and Jean H. Laherrere. 1998. The end of cheap oil. *Scientific American*. (Mar): 60–65.

Carrington, C. Gerald. 1994. *Basic thermodynamics*. Oxford: Oxford University Press. ISBN 0-19-851747-5 pbk.

Casti, John L. 1996. *Five golden rules : great theories of 20th-century mathematics and why they matter*. New York: John Wiley and Sons. ISBN 0-471-19337-2 pbk.

Casti, John L. 2000. *Five more golden rules : knots, codes, chaos, and other great theories of 20th-century mathematics*. New York: John Wiley and Sons. ISBN 0-471-32233-4.

Chung, W., Y. June Wu, and J. David Fuller. 1997. Dynamic energy and environment equilibrium model for the assessment of CO_2 emission control in Canada and the USA. *Energy Economics*. **19**(1): 103–124.

Clayton, Anthony M.H. and Nicholas J. Radcliffe. 1996. *Sustainability : a systems approach*. London: Earthscan. ISBN 0-85383-319-3.

Cocklin, Christopher R. 1989. Metholodogical problems in evaluating sustainability. *Environmental Conservation*. **16**(4): 343–351.

Coveney, Peter V. 1988. The second law of thermodynamics : entropy, irreversibility and dynamics. *Nature*. **333**: 409–415. June.

Curry, Dave A. 1996. UNIX *system programming for SVR4*. Sebastopol, California: O'Reilly and Associates. ISBN 1-56592-163-1.

Daellenbach, Hans G., John A. George, and Donald C. McNickle. 1983. *Introduction to operations research techniques*. Second edition. Boston: Allyn and Bacon. ISBN 0-205-07718-8.

Dang, Hien D.T. 1999. *New Zealand energy data file — July 1999*. Wellington, New Zealand: Energy Modelling and Statistics Unit, Ministry of Commerce. July.

De Vries, H.J.M., J.G.J. Olivier, R.A. van den Wijngaart, G.J.J. Kreileman, and A.M.C. Toer. 1994. Model for calculating regional energy use, industrial production and greenhouse gas emissions for evaluating global climate scenarios. *Water Air and Soil Pollution*. **76**: 79–131.

Derigs, Ulrich. 1988. *Programming in networks and graphs : on the combinatorial background and near-equivalence of network flow and matching algorithms*. Berlin: Springer-Verlag. ISBN 3-540-18969-6.

Deverson, Tony (ed). 1997. *The New Zealand pocket Oxford dictionary*. Second edition. Auckland, New Zealand: Oxford University Press. ISBN 0-19-558379-5.

Dolan, Alan and Joan Aldous. 1995. *Networks and algorithms : an introductory approach*. Chichester, England: John Wiley and Sons. ISBN 0-471-93993-5 pbk. [Reprinted with corrections, first issued 1993].

Dovers, Stephen R. and John W. Handmer. 1993. Contradictions in sustainability. *Environmental Conservation*. **20**(3): 217–222.

Edison Design Group. 1998. C++ front end : internal documentation — Version 2.38. Software documentation. Edison Design Group. New Jersey. 13 January.

Ehrlich, P.R. and J.P. Holdren (eds). 1988. *Cassandra conference : resources and the human predicament*. College Station, Texas: Texas A & M University Press.

Ellis, Margaret A. and Bjarne Stroustrup. 1992. *The annotated C++ reference manual*. Reading, Massachusetts: Addison-Wesley. ISBN 0-201-51459-1. [Reprinted with corrections, first issued 1990].

Fang, Shu-Cherng, J.R. Rajasekera, and H.-S.J. Tsao. 1997. Entropy optimization and mathematical programming. *Kluwer Academic Publishers*.

Faucheux, Sylvie, Martin O'Connor, and Jan van der Staaten. 1998. *Sustainable development : concepts, rationalities and strategies*. Dordrecht: Kluwer Academic Publishers. ISBN 0-7923-4884-2.

Forrester, Jay Wright. 1965. *Industrial dynamics*. Cambridge, Massachussets: MIT Press.

Foulds, L.R. 1979. *A multi-commodity flow network design problem — Massey University Occassional Publication in Mathematics no 8*. Palmerston North, New Zealand: Department of Mathematics, Massey University.

Foulds, L.R. 1981. *Optimization techniques : an introduction*. New York: Springer-Verlag.

Foulds, L.R. 1994. *Graph theory applications*. Berlin: Springer-Verlag. ISBN 0-387-975993. [Corrected second printing, first issued 1992].

Fuge, Paul, Alison Little, and Bronwyn Ward. 2000. Impact of distributed generation on New Zealand's transmission system. In: *Proceedings, EEA Annual Conference 2000*. pp. 77–85. Held at Sheraton Hotel, Auckland, New Zealand, 16–17 June 2000.

Fulkerson, D.R. 1961. An out-of-kilter method for minimal-cost flow problems. *SIAM Journal on Applied Mathematics*. **9**: 18–27.

Gibbons, Alan M. 1985. *Algorithmic graph theory*. Cambridge: Cambridge University Press. ISBN 0-521-28881-9 pbk.

Gilly, Daniel. 1992. UNIX *in a nutshell : System V edition*. Second edition. Sebastopol, California: O'Reilly and Associates. ISBN 1-56592-001-5.

Glasby, G.P. 1991. A review of the concept of sustainable management as applied to New Zealand. *Journal of the Royal Society of New Zealand*. **21**(2): 61–81.

Golden, B.L. and T.L. Magnanti. 1977. Deterministic network optimization : a bibliography. *Networks*. **7**: 149–183.

Goodheart, Berny and James Cox. 1994. *The magic garden explained : the internals of the* UNIX *System V Release 4 : an open systems design*. New Jersey: Prentice Hall. ISBN 0-13-098138-9.

Groscurth, Helmuth-M., Thomas Bruckner, and Reiner Kümmel. 1993. Energy, cost, and carbon dioxide optimization of disaggregated, regional energy-supply systems. *Energy*. **18**(12): 1187–1205.

Groscurth, Helmuth-M., Thomas Bruckner, and Reiner Kümmel. 1995. Modeling of energy-services supply systems. *Energy*. **20**(9): 941–958.

Groscurth, Helmuth-M. and K.-P. Kress. 1998. Fuzzy data compression for energy optimization models. *Energy*. **23**(1): 1–9.

Hannon, Bruce, Matthais Ruth, and Evan Delucia. 1993. A physical view of sustainability. *Ecological Economics*. **8**: 253–268.

Harvey, Scott M. and William W. Hogan. 2000. Nodal and zonal congestion and the exercise of market power. 10 January. Unpublished paper. [Available from web URL: `http://ksgwww.harvard.edu/people/whogan`].

Hawken, Paul, Amory B. Lovins, and L. Hunter Lovins. 1999. *Natural capitalism : the next industrial revolution*. London: Earthscan. ISBN 1-85383-461-0.

Hendtlass, C.A., N. John Peet, and James T. Baines. 1988. *Energy analysis of goods and services in New Zealand — Report no 162*. Wellington, New Zealand: New Zealand Energy Research and Development Committee (NZERDC). June.

Henning, Dag. 1997. MODEST : an energy-system optimisation model applicable to local utilities and countries. *Energy*. **22**(12): 1135–1150.

Henning, Dag. 1998. Cost minimization for a local utility through CHP, heat storage and load management. *International Journal of Energy Research*. **22**(8): 691–713.

Hogan, William W., Brendan J. Ring, and E. Grant Read. 1995. *Using mathematical programming for electricity spot pricing — Working paper EMRG–WP–95–03*. Christchurch, New Zealand: Department of Management, University of Canterbury.

Howarth, Richard B. and Richard B. Norgaard. 1995. Intergenerational choices under global environmental change. pp. 111–137. In: *Handbook of environmental economics*. Editor: Bromley, Daniel W. Cambridge, Massachusetts: Blackwell. ISBN 1-55786-641-4 pbk.

Illingworth, Valerie (ed). 1996. *Dictionary of computing*. Fourth edition. Oxford: Oxford University Press. ISBN 0-19-853855-3.

Intergovernmental Panel on Climate Change. 1994. *Radiative forcing of climate change : the 1994 report of the Scientific Assessment Working Group of IPCC : summary for policymakers*. Intergovernmental Panel on Climate Change.

Jackson, Tim. 1996. *Material concerns : pollution, profit, and quality of life*. London: Routledge. ISBN 0-415-13249-5.

Jantsch, Erich. 1980. *The self-organizing universe : scientific and human implications of the emerging paradigm of evolution*. Oxford: Pergamon Press. ISBN 0-08-024311-8 pbk.

Jaynes, Edwin J. 1979. Where do we stand on maximum entropy?. pp. 15–118. In: *The maximum entropy formalism*. Editors: Levine, Raphael D. and Myron Tribus. Cambridge, Massachusetts: MIT Press. ISBN 0-262-12080-1.

Jensen, P.A. and J.W. Barnes. 1980. *Network flow programming*. New York: John Wiley and Sons. ISBN 0-471-04471-7.

Jensen, Randall W. and Bruce O. Watkins. 1974. *Network analysis : theory and computer methods*. Englewood Cliffs, New Jersey: Prentice-Hall. ISBN 0-13-611061-4.

Jewell, William S. 1962. Optimal flow through networks with gains. *Operations Research*. **10**: 476–499.

Kabelac, S. and F.D. Drake. 1992. The entropy of terrestrial solar radiation. *Solar Energy*. **48**(4): 239–248.

Kask, S.B. 1988. *A conceptual framework for energy policy analysis — Report no 147*. Wellington, New Zealand: New Zealand Energy Research and Development Committee (NZERDC). March.

Keenan, J.H. 1932. Steam chart for second-law analysis : study of thermodynamic availability in the steam-power plant. *Mechanical Engineering the Journal of the American Society of Mechanical Engineers*. **54**: 195–204. March.

Kennington, J. and R.V. Helgason. 1980. *Algorithms for network programming*. New York: John Wiley and Sons.

Kittel, Charles and Herbert Kroemer. 1980. *Thermal physics*. Second edition. New York: W.H. Freeman. ISBN 0-7167-1088-9.

Koomey, Jonathan G., R. Cooper Richey, Skip Laitner, Robert J. Markel, and Chris Marnay. 1998. Technology and greenhouse gas emissions : an integrated scenario analysis using the LBNL-NEMS model — LBNL–42054/UC–000/EPA 430–R–98–021. Report. Lawrence Berkeley Laboratory & US Environmental Protection Agency Office of Atmospheric Programs. USA. September. [Available from web URL: `http://enduse.lbl.gov/Projects/GHGcosts.html`].

Kotas, Tadeusz Jozef. 1995. *The exergy method of thermal plant analysis*. Reprinted with corrections and new appendix G edition. Malabar, Florida: Krieger Publishing. ISBN 0-89464-946-9.

Krakow, Kalman I. 1991. Exergy anaylsis : dead-state definition. *ASHRAE Transactions: Research*. **97**(pt.1): 328–337.

Krakow, Kalman I. 1994. Relationships between irreversibilty, exergy destruction, and entropy generation for systems and components. *ASHRAE Transactions Research*. **100**(pt.1): 3–10.

Kümmel, Reiner. 1989. Energy as a factor of production and entropy as a pollution indicator in macroeconomic modelling. *Ecological Economics*. **1**: 161–180.

Kümmel, Reiner and Uwe Schüßler. 1991. Heat equivalents of noxious substances : a pollution indicator for environmental accounting. *Ecological Economics*. **3**: 139–156.

Kydes, Andy S., Susan H. Shaw, and Douglas F. McDonald. 1995. Beyond the horizon : recent directions in long-term energy modeling. *Energy*. **20**(2): 131–149.

Lawler, Eugene L. 1989. Combinatorial structures and combinatorial optimization. pp. 162–197. In: *Combinatorial optimization*. Editor: Simeone, B. Berlin: Springer-Verlag. ISBN 3-540-51797-9.

Lehey, Greg. 1995. *Porting* UNIX *software : from download to debug*. Sebastopol, California: O'Reilly and Associates. ISBN 1-56592-126-7.

Lehtila, A. and P. Pirila. 1996. Reducing energy related emissions : using an energy systems optimization model to support policy planning in Finland. *Energy Policy*. **24**(9): 805–819.

Lenssen, Nicholas and Christopher Flavin. 1996. Sustainable energy for tomorrow's world : the case for an optimistic view of the future. *Energy Policy*. **24**(9): 769–781.

Lewine, Donald A. 1991. *POSIX programmer's guide : writing portable* UNIX *programs*. Sebastopol, California: O'Reilly and Associates. ISBN 0-937175-73-0.

Lindenberger, Dietmar, Thomas Bruckner, Helmuth Groscurth, and Reiner Kümmel. 2000. Optimization of solar district heating systems : seasonal storage, heat pumps, and cogeneration. *Energy*. **25**(7): 591–608.

Linnhoff, Bobo. 1993. Pinch analysis : a state-of-the-art overview. *Chemical Engineering Research and Design*. **71**(A5): 503–522.

Linnhoff, B., D.W. Townsend, D. Boland, G.F. Hewitt, B.E.A. Thomas, A.R. Guy, and R.H. Marsland. 1982. *User guide on process integration for the efficient use of energy*. Institution of Chemical Engineers, UK. ISBN 0-85295-156-6.

Lippman, Stanley B. 1995. *C++ primer*. Second edition. Reading, Massachusetts: Addison Wesley. ISBN 0-201-54848-8. [Reprinted with corrections, first issued 1991].

Luenberger, David G. 1979. *Introduction to dynamic systems : theory, models, and applications*. New York: John Wiley and Sons. ISBN 0471025941.

Marcella, Thomas V. 1992. Entropy production and the second law of thermodynamics : an introduction to second law analysis. *American Journal of Physics*. **60**(10):888–895.

Marganti, T.L. and R.T. Wong. 1984. Network design and transport planning : models and algorithms. *Transportation Science*. **18**(1): 1–55.

Matsuhashi, R. and H. Ishitani. 1995. Model analyses for sustainable energy supply under CO_2 restrictions. *IEEE Transactions on Energy Conversion*. **10**(4): 730–735.

May, Robert. 1986. When two and two do not make four : nonlinear phenomena in ecology. *Proceedings of the Royal Society*. **B228**: 241–266.

McMullan, John T. 1999. Refrigeration and the environment : issues and strategies for the future. In: *20th International Congress of Refrigeration, IIR/IIF*. Held Sydney, Australia, 1999.

Meadows, Donella, Dennis Meadows, Jorgen Randers, and William Behrens. 1974. *The limits to growth : a report for the Club of Rome's project on the predicament of mankind*. Second edition. New York: Universe Books. ISBN 0-87663-918-X pbk.

Meadows, Donella H., Dennis L. Meadows, and Jorgen Randers. 1992. *Beyond the limits : global collapse or a sustainable future*. London: Earthscan.

Mehlhorn, Kurt and Stefan Näher. 1999. LEDA *: a platform for combinatorial and geometric computing*. Cambridge: Cambridge University Press. ISBN 0-521-56329-1.

Messner, S., A. Golodnikov, and A. Gritsevskii. 1996. A stochastic version of the dynamic linear programming model MESSAGE III. *Energy*. **21**(9): 775–784.

Ministry for the Environment. 1997. *The state of New Zealand's environment 1997*. Wellington, New Zealand: Ministry for the Environment. ISBN 0-478-09000-5.

Ministry for the Environment. 1999. *Climate change : domestic policy options statement : a consultation document*. Wellington, New Zealand: Ministry for the Environment. January. ISBN 0-478-09052-8.

Ministry of Commerce. 1992. *A pinch, and a punch ... for fertiliser production efficiency — Demonstration project summary 26*. Wellington, New Zealand: Energy Management, Ministry of Commerce. June.

Ministry of Commerce. 1994. *Technical guidelines for establishing and reporting voluntary agreements to reduce carbon dioxide emissions in New Zealand — First issue*. Wellington, New Zealand: Ministry of Commerce. December.

Ministry of Commerce. 1999. *Energy greenhouse gas emissions 1990–1998*. Wellington, New Zealand: Energy Markets Information and Services, Ministry of Commerce. June.

Ministry of Commerce. 2000. *New Zealand energy outlook to 2020 : a report prepared by the Ministry of Commerce Energy Modelling and Statistics Unit, Resources and Networks Branch*. Wellington, New Zealand: Ministry of Commerce.

Morrison, Robbie. 1998. A network view of sustainable energy systems. Unpublished paper presented at the New Zealand Wind Energy Association Keith Dawber Memorial Seminar, Dunedin, New Zealand, 27 July 1998.

Morrison, Robbie. 1999. A network view of sustainable (energy-services) systems. In: *Threshold 2000 : can our cities become sustainable — Proceeding of the 6th conference of the Sustainable Energy Forum*. Editor: Mander, Neil K. pp. 87–90. Held at Auckland University, Auckland, New Zealand. 23–25 June. ISBN 0-958-3472-3-9.

Morrison, Robbie. 2000. Systematic energy system integration for improved sustainability. In: *Proceedings, EEA Annual Conference 2000*. pp. 195–201. Held at Sheraton Hotel, Auckland, New Zealand, 16–17 June 2000. ISSN 1174-619X.

National Business Review (Auckland). 1998. 'Hydro royalty' call in Contact Energy sale. p. 11. 16 October.

Nemhauser, George L. and Laurence A. Wolsey. 1988. *Integer and combinatorial optimization*. New York: John Wiley and Sons. ISBN 0-471-82819-X.

Nicholson, Walter. 1995. *Microeconomic theory : basic principles and extensions*. 6th edition. Fort Worth: The Dryden Press. ISBN 0-03-007554-8.

Nieuwlaar, Evert. 1996. ENERPAC 5 : a package for the thermodynamic analysis of energy systems — Fifth edition / Report number 96057 / Software. Source code and manual. Department of Science, Technology and Society, Utrecht University. Utrecht, The Netherlands. 13 August.

Odum, Howard T. 1994. The emergy of natural capital. pp. 200–214. In: *Investing in natural capital : the ecological economics approach to sustainability*. Editors: Jansson, A., M. Hammer, C. Folke, and R. Constanza. Washington: Island Press. ISBN 1-55963-316-6.

Ore, Oystein. 1990. *Graphs and their uses*. Revised and updated by Robin J. Wilson edition. Washington DC: The Mathmatical Association of America. ISBN 0-88385-635-2.

Ossebaard, Marjan E., Ad J.M. van Wijk, and Mark T. van Wees. 1997. Heat supply in the Netherlands : a systems analysis of costs, exergy efficiency, CO_2 and NO_x emissions. *Energy*. **22**(11): 1087–1098.

Parikh, Jyoti K. 1998. The emperor needs new clothes : long-range energy-use scenarios by IIASA–WEC and IPCC. *Energy*. **3**(1): 69–70.

Parliamentary Commissioner for the Environment. 1998. *Towards sustainable development : the role of the Resource Management Act 1991 — PCE Environmental Management Review no 1*. Wellington, New Zealand: Parliamentary Commissioner for the Environment. August.

Parliamentary Commissioner for the Environment. 2000. *Getting more from less : a review of progress on energy efficiency and renewable energy initiatives in New Zealand*. Wellington, New Zealand: Parliamentary Commissioner for the Environment. February. ISBN 0-908804-90-3.

Pass, Christopher, Bryan Lowes, and Leslie Davies. 1988. *Economics : Collins reference dictionary*. London: Collins. ISBN 0-00-434353-0.

Patterson, Murray G. 1987. *A systems approach to energy quality and efficiency — Report no 143*. Wellington, New Zealand: New Zealand Energy Research and Development Committee (NZERDC). June.

Patterson, Murray G. 1993. *Energy efficiency monitoring : an evaluation of possible indicators and implementation options*. Wellington, New Zealand: Energy Efficiency and Conservation Authority. ISBN 0-478-00294-7.

Patterson, Walt. 1999. *Transforming electricity : the coming generation of change*. London: The Royal Institute of International Affairs & Earthscan. ISBN 1-85383-341-X.

Pearce, Fred. 1998. Consuming myths : conserving energy saves people money, but as a solution to the problem of global warming it may well be fatally flawed. *New Scientist*. (05 Sept): 18–19.

Pearson, R.G. 1977. *Energy analysis — Report no 30*. Wellington, New Zealand: New Zealand Energy Research and Development Committee (NZERDC).

Peet, John. 1992. *Energy and the ecological economics of sustainability*. Washington DC: Island Press. ISBN 1-55963-160-0 pbk.

Peet, John and James Baines. 1986. *Energy analysis : a review of theory and applications — Report no 126*. Auckland, New Zealand: New Zealand Energy Research and Development Committee (NZERDC).

Peusner, Leonardo. 1986. *Studies in thermodynamic networks*. Amsterdam: Elsevier. ISBN 0-444-42580-2.

Press, William H., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1993. *Numerical recipes in C : the art of scientific computing*. Second edition. Cambridge: Cambridge University Press. ISBN 0521431085.

Prigogine, Ilya and Isabelle Stengers. 1984. *Order out of chaos : man's new dialogue with nature*. Toronto: Bantam Books.

Psarras, J., P. Capros, and J.-E. Samouilidis. 1990. Multiobjective programming. *Energy*. **15**(7/8): 583–605.

Raymond, Eric S. 1998a. The cathedral and the bazaar — v1.38. 13 May. Internet document.

Raymond, Eric S. 1998b. Open source software : a (new?) development methodology — v1.3. 08 November. Internet document.

Read, E.Grant. 1997. Transmission pricing in New Zealand. *Utilities Policy*. **6**(3): 227–235.

Read, E. Grant. 1998. *Electricity sector reform in New Zealand : lessons from the last decade — Working paper EMRG–WP–98–02*. Christchurch, New Zealand: Department of Management, University of Canterbury.

Read, E. Grant, Glenn R. Drayton-Bright, and Brendan J. Ring. 1998. *An integrated energy and reserve market for New Zealand — Working paper EMRG0–WP–98–01*. Christchurch, New Zealand: Department of Management, University of Canterbury.

Robert, Karl-Henrik, John Holmberg, and Karl-Erik Eriksson. 1995. *Socio-ecological principles for a sustainable society : scientific background and Swedish experience*. Stockholm: The Natural Step Environmental Institute.

Rosen, M.A. 1999. Second-law analysis : approaches and implications. *International Journal of Energy Research*. **23**: 415–429.

Rosen, Marc A. and Ibrahim Dincer. 1997. On exergy and environmental impact. *International Journal of Energy Research*. **21**: 643–654.

Rosenblatt, Bill. 1993. *Learning the Korn Shell*. Sebastopol, California: O'Reilly and Associates. ISBN 1-56592-054-6.

Rossouw, Pieter, Jonathan Lermit, and Barry James. 1997. Energy efficiency resource assessment as a tool for quantifying past, present and future energy efficiency uptake rates and potential. In: *Proceedings, IPENZ Annual Conference 1997*. Volume 2. pp. 164–167. Intitution of Professional Engineers New Zealand Inc, Wellington, New Zealand. February. ISBN 0-908960-20-4.

Ruth, Matthias. 1993. *Integrating economics, ecology and thermodynamics*. Dordrecht: Kluwer Academic Publishers. ISBN 0-7923-2377-7.

Scott, T.J. and E. Grant Read. 1995. *Modelling hydro reservoir operation in a deregulated market — Working paper EMRG–WP–95–05*. Christchurch, New Zealand: Department of Management, University of Canterbury.

Seborg, Dale E., Thomas F. Edgar, and Duncan A. Mellichamp. 1989. *Process dynamics and control*. New York: John Wiley and Sons. ISBN 0471863890.

Sedgewick, Robert. 1983. *Algorithms*. Reading, Massachusetts: Addison-Wesley. ISBN 0-201-06672-6.

Slattery, John Charles. 1981. *Momentum, energy, and mass transfer in continua*. Second edition. New York: R.E. Krieger.

Smith, Brian R. 1978. *An optimisation model of the NZ energy supply and distribution system*. Wellington, New Zealand: New Zealand Energy Research and Development Committee (NZERDC). [A short summary has been published separately].

Smith, David Kendall. 1982. *Network optimisation practice : a computational guide*. Chichester: Ellis Horwood. ISBN 0-85312-403-5.

Stepanov, V.S. 1995. Chemical energies and exergies of fuels. *Energy*. **20**(3): 235–242.

Stevens, Richard W. 1992. *Advanced programming in the* UNIX *environment*. Reading, Massachusetts: Addison-Wesley.

Stroustrup, Bjarne. 1991. *The C++ programming language*. Second edition. Sebastopol, California: Addison-Wesley Publishing. ISBN 0-201-53992-6. [This edition more closely aligns with the language standards used for *deeco* than does the third edition].

Sun, Zhifa F. and C. Gerald Carrington. 1991. Application of nonequilibrium thermodynamics in second law analysis. *Journal of Energy Resources Technology*. **133**: 33–39.

The Economist. 1995. At the going down of the nuclear sun. pp. 97–100. 16 September.

Tribus, Myron. 1979. Thirty years of information theory. pp. 1–10. In: *The maximum entropy formalism*. Editors: Levine, Raphael D. and Myron Tribus. Cambridge, Massachusetts: MIT Press. ISBN 0-262-12080-1.

United Nations Environment Program. 1999. *Global environment outlook 2000*. London: Earthscan. ISBN 1-85383-588-9 pbk.

UNIX Systems Laboratories. 1992. *Programming with* UNIX *system calls :* UNIX *SVR4.2*. Englewood Cliffs, New Jersey: UNIX Press.

Upton, Simon. 1995. Air discharge permit : Taranaki combined cycle power station : decision of Hon. Simon Upton, Minister for the Environment. 23 March.

van Gool, Willem. 1987. The value of energy carriers. *Energy*. **12**: 509–518.

van Pelt, M., A. Kuyvenhoven, and P. Nijkamp. 1990. *Project appraisal and sustainability : the applicability of cost-benefit and multi-criteria analysis*. Wageningen Agricultural University.

von Weizäcker, Ernst, Amory B. Lovins, and Hunter L. Lovins. 1997. *Factor four : doubling wealth — halving resource use : a new report to the Club of Rome*. Sydney: Allen and Ulwin.

Wall, Göran. 1998. Exergetics. Draft manuscript. [Available from web URL: `http://exergy.se`].

Wall, Larry, Tom Christiansen, Randal L. Schwartz, and Stephen Potter. 1996. *Programming* PERL. Second edition. Sebastopol, California: O'Reilly and Associates. ISBN 1-56592-149-6.

Ward, Mark. 1998. There's an ant in my phone. *New Scientist*. (24 Jan): 32–35.

Weber, Bruce H., David J. Depew, and James D. Smith (eds). 1988. *Entropy, information and evolution : new perspectives on physical and biological evolution*. Cambridge, Massachusetts: MIT Press.

Weitzen, T.C. 1992. The C++ graph classes : a tutorial. pp. 7/1–7/30. In: *USL C++ Standard Components, release 3.0, programmers guide*. Editor: AT&T & UNIX Systems Laboratories. AT&T & UNIX Systems Laboratories. [Part of the Hewlett-Packard HP-UX workstation documentation].

West, Geoffrey, James H. Brown, and Brian J. Enquist. 1997. A general model for the origin of allometric scaling laws in biology. *Science*. **276**: 122–126.

Williams, David A.R., David Elms, and Noel Johnston. 1995. Proposed Taranaki power station : air discharge effects : report and recommendations pursuant to section 148 of the Resource Management Act 1991. February.

Winter, Pawel. 1989. Topological network synthesis. pp. 282–303. In: *Combinatorial optimization*. Editor: Simeone, B. Berlin: Springer-Verlag. ISBN 3-540-51797-9.

WOGOCOP. 1996. Climate change and CO_2 policy : a durable response. Discussion document of the working group on CO_2 policy. Ministry for the Environment. Wellington. June. ISBN 0-478-09003-X. [WOGOCOP is Working Group on the Conference of the Parties].

Yaged, B. 1973. Minimum cost routing for dynamic network models. *Networks*. **3**: 193–224.

Yakovleva, M.A. 1959. Problem of minimum transportation expense. pp. 390–399. In: *Applications of Mathematics to Economic Research*. Editor: Nemchinov, V.S. Moscow, USSR.

Yang, Miao and E. Grant Read. 1999. *A constructive dual DP for a reservoir model with correlation — Working paper EMRG–WP–99–01*. Christchurch, New Zealand: Department of Management, University of Canterbury.

Yasni, Eli and C. Gerald Carrington. 1988. Off-design exergy audit of a thermal power station. *Journal of Engineering for Gas Turbines and Power*. **110**: 166–172. [Transactions of the American Society for Mechanical Engineering].

Zarnikau, Jay, Sid Guermouches, and Philip Schmidt. 1996. Can different energy resources be added or compared? *Energy*. **21**(6): 483–491.

Ziemba, W.T., S.L. Schwartz, and Ernest Koenigsberg. 1980. *Energy policy modeling : United States and Canadian experiences*. Boston: M. Nijhoff.

This thesis was prepared using LaTeX 2ε ⟨1996/06/01⟩ on Linux 2.0.33. Referencing was managed using BibTeX, the package *natbib* v6.8 (for author-date citation), and a modified *makebst* `.bst` file. The references were databased in MS Access 97 and written out in `.bib` format using Visual Basic (VBA) subroutines. Diagrams were prepared with Adobe Illustrator 7.0 and imported as encapsulated PostScript (EPS). The final document was output as level 2.0 PostScript using *dvips* 5.58f and printed using a 600 dpi laser printer.

Although LaTeX has a substantial coding overhead, its equation setting, cross-referencing, and unparalleled stability make it well-suited for preparing documents of this type. Much of the software used was open-source (OSS) or similar — I therefore owe a debt of gratitude to the open source community for these excellent products. □