Combinatorial Systems and Schema and their Cohomology with Applications in Organizational Systems.

ANDREW MACFARLANE*

March 26, 2013

Abstract

The category of combinatorial systems is a model category for "scaling up" algorithms. This involves networks of algorithms, herein called schedules, acting on very wide range of inputs types that are outputs from other algorithms. This patching algorithms over networks gives rise to the subcategory of combinatorial modules which are an algebraic analogy for the creation of manifolds from Euclidean spaces. Combinatorial systems model large-scale systems including precision manufacturing networks. Schedules have both local and global significance and form group valued sheaves, the Abelian category of which is the combinatorial schema for a given combinatorial system. It contains the data on the dynamics and capability of the system with individual sheaves corresponding to dynamic conditions. A finite network adaptation of Čech cohomology is used to characterize the sheaves. A given sheaf of schedules has a large class of related sheaves of measures, groupoid actions etcetera. The cohomology classes of these sheaves have significant interpretations in manufacturing networks such as component substitutions and operating conditions. For example, the cohomology of groupoid actions applies to rescheduling in networks of factories. In other cases cohomology tracks the change in capability to handle excessive order loads.

Keywords: Combinatorial systems; sheaves of schedules, measures and groupoids; Abelian categories of sheaves; cohomology; manufacturing networks.

2010 Mathematics Subject Classification: 08A99, 18F20, 18G60, 93A30.

1 Introduction

This paper introduces the category of combinatorial systems as systems that are intended to model a range of organizational and natural structures but also to model the working of algorithms in those structures. Thus they are intermediate between processes that can be done in social and natural contexts and the theoretical description of those processes as algorithms with many heterogeneous inputs. In this way the paper extends the concept of mathematical algorithm as a process of calculation

*Email: combinatorial.schema@gmail.com
to the physical realm of engineering processes. However the paper makes no contribution to the mathematics of algorithms, Turing machines and their generalizations [28] which is a different area of (extensive) research. Instead the starting point is the observation that many advanced technology products be they one off, such as satellite or space probes, or produced in quantity, such as smart phones, solid state memory or jet turbines, are manufactured in the most precise and defined ways. It is only by an algorithmic approach to production engineering that the highest quality can be achieved. If we accept that algorithms

1. accept a range of inputs satisfying a range of properties and, by a definite set of transformations, "bring into being" a desire output that satisfies a set of specifications;

2. when all inputs are within specification, all steps followed as prescribed, the result is assured,

then certain physical and engineering processes deserve the name "algorithm". For the mathematician it is only necessary that the algorithm is designed to do such a thing. Errors bedevil all processes and assurance cannot be given. Assurance belongs to the theory of the algorithm. Metabolic chains are theoretical algorithms but their actual workings depends on their environment which can alter the outcome. Large organizations and natural processes exist by integrating the assured processes within them. This means we must include concepts of "local" and "global" and introduce techniques to investigate these integration structures. Such techniques are drawn from algebraic topology. The intended applications of combinatorial systems is the dynamics of large-scale finite structures with states that are far from homogeneous. We start by giving a sample of such structures. Some are subject to ongoing research, others have aspects of integration that seem to the author to fall within the purview of this paper.

1. "Combinatorial modules:" Let $A$ be a directed set of finitely generated non-commutative rings acting on state spaces, for each $a \in A$, $W_a$ is a ring and $Y_a$ a state space with action

$$\alpha_a : W_a \times Y_a \rightarrow Y_a.$$ 

If $b$ is covered by $a$ in $A$, $Y_a \cap Y_b \neq \emptyset$ so the partial order in $A$ controls an exchange of information among the state spaces and produces a form of dynamics. This algebraic construction has applications in most of the systems below. In particular, algorithms can frequently be cast as monoids of actions acting on a state space. Monoids of operations or actions can generate non-commutative rings which act on modules generated by states.

2. Large-scale information systems such as weather reporting systems with multiple graphics and images. If such systems are large-scale recursive functions then they have thousands of inputs (or "oracles" in recursive function language) refreshing data regularly. There are many techniques for arranging such systems into a network of processing nodes [34]. These typically process data where it was received and then send the results to the next layer of processing for further integration and processing for the final displays or reports.
3. All manufacturing that requires precision, especially highly technical manufacturing, even with one-off items such as a space probe or new telecommunication satellite, requires careful planning and in many cases is a "engineering algorithm". Furthermore the planning expands to all suppliers. Precisely controlled manufacturing and testing processes make an often minutely defined, extended manufacturing algorithm. Volume manufacturing of high technology consumer products cannot be successful without the production and supplier environment that is again an engineering algorithm. This area of research is now called supply chain science [33]. The point of view here is that manufacturing networks (supply chains) represent the physically manifestation of algorithms on a grand scale.

4. The actual testing of large-scale systems mimics the system itself but has quite different dynamics and structure [23].

5. Multiple-scale systems. These are systems for which the calculation of properties for the entire set requires calculations that aggregate or integrate effects from the smallest scale up through intermediate scales (mesoscale) to the whole system. In this case we are interested in systems for which there is a finite set of minimum scales and all intermediate scale systems are built from subsets of systems on the preceding scale. Examples include finite element simulations for systems with heterogeneous materials. Examples in natural and social systems are not obviously "algorithmic" being stochastic in nature. Typically we attempt to understand the interactions initially through averaged interactions and the apply a Monte-Carlo randomization to find the breadth of possible outcomes. Examples include:

(a) Ecosystems. Each organism in the system can have life stages that require particular resources. These resources are supplied by members of the same species and by other fauna or flora of the ecosystem. The approach is different from dynamic systems of species populations and will be explained when all the necessary concepts are in place.

(b) Recovery problems wherein the recovery of a target system such as a forest or an economic system requires the combinations of subsystems to be brought "online" and to create higher level interactions. The recovery times can be months or years and the failure to integrate is a significant problem.

(c) Metabolic networks in which cellular level processes produce needed biochemicals for other cells in an animal’s response to environmental stimuli (http://en.wikipedia.org/wiki/Metabolism).

6. Modeling flows such as river systems that aggregate branching flows from many sources to a single sink. Such flows scale up fluid dynamics. The structure of flows are dependent on local phenomena and their analysis analogous to transport problems.

7. Many of these examples model structures that can work in reverse order: instead of concentrating components and or information they disperse it. Thus distribution of humanitarian aid is a supply network in reverse. A multilayer system in reverse is a series of local adjustments that take place in the context of maintaining an overall "global" system.
All of these examples are examples of systems or collections of ideas, states and objects that are heterogeneous and interact or contribute on various scales. They all have a sense of progress towards "aggregated" states and so have a definite sense of dynamics. These systems are examples of combinatorial systems which are now introduced.

The contribution of this paper is to provide analytic tools to investigate classes of systems that can be seen as the model categories of algorithms. Such systems have their own type of dynamics that has significant economic importance.

The paper is organized into two parts. Part 1 introduces all the concepts and gives the initial theorems. Part 2 illustrates their interpretation and use in manufacturing networks.

Part I

Combinatorial Systems and Schema

2 Definitions and concepts

Let $\text{Cat}$ be a 2-category of small categories with pullbacks.

A combinatorial system is a 2-category $\mathcal{A} = (A, Tpc : A \to \text{Cat})$ where $A$ is a finite directed graph and so a category. $Tpc(a)$ is the "topic a". Each category $Tpc(a)$ will have a set of starting objects that are not the range of any maps in $Tpc(a)$ and a set of final objects that are not the domain of any maps in $Tpc(a)$. We use $e, e', e_1, e_2, ..., e_n$ for objects in a category $Tpc(a)$ which we shall call "types" (to cover objects from mathematics and computer science, concepts, definitions, states, components, configurations, species ...) and $w, w', w_1, w_2, ..., w_n$ for maps in $Tpc(a)$.

1. For any pair $a_1, a_2 \in A$ there is one or more $a$ that covers them. The set of predecessors of $a$, $\{ b \mid b \prec a \}$ is the set $S(a)(= S^1(a))$ and the successors of $a$ are $S^+(a)(= S^{+1}(a))$.

2. $a_1 \in S(a_2)$ if and only if there is a non-empty set $\tilde{\beta}(a_2, a_1) \subset \text{Obj}(Tpc(a_1)) \cap \text{Obj}(Tpc(a_2))$ such that if $e$ is an object in $\tilde{\beta}(a_2, a_1)$ then $e$ is a final object in $Tpc(a_1)$ (appears only in the range of maps) and a starting object in $Tpc(a_2)$ (appears only in the domain of maps).

3. For each $a \in A$ there is a function $\theta(a) : \text{Obj}(Tpc(a)) \to \mathbb{N}$ (or $\mathbb{Q}$, $\mathbb{R}$) called a type weighting such that if $w : e_1 \to e_2$ then $\theta(a)(e_2) \geq \theta(a)(e_1)$.

$\tilde{\beta}(a_2, a_1)$ is the overlap of $Tpc(a_1)$ and $Tpc(a_2)$ and is the way the combinatorial system connects to become a single structure. The elements of $A$ are written opposite to the ordering because in the applications studied below $\tilde{\beta}$ is associated with a "requesting" function $\beta(a_2, a_1) : \text{Obj}(Tpc(a_2)) \to \text{Obj}(Tpc(a_1))$ which request from $a_1$ "components" for objects in $Tpc(a_2)$.

The following notation is used throughout this paper.

$$S^{n+1}(a) = \{ b \mid \exists b' \in S^n(a) \land b \in S(b') \}$$

and
Every lead-time function is associated with a type weighting and Proposition 2.1.

Given a lead-time function \( L : \text{Maps}(\text{Tpc}(a)) \rightarrow \mathbb{N} \)

such that \( L(w_2 \circ w_1) = L(w_2) + L(w_1) \). For \( a \in A \) ”lead-time through \( a \)” is max \( \{ L(w) \mid w \in \text{Maps}(\text{Tpc}(a)) \} \).

**Proposition 2.1.** Every lead-time function is associated with a type weighting and vice versa.

**Proof.** Given a lead-time function \( L, \) if \( e = w(e') \) define a type weighting

\[
\theta(a)(e) = \max \{ L(w) \mid w(e') = e \}.
\]

Given a type weighting \( \theta \) define a lead-time function by

\[
L(w) = \max \{ \theta(e) - \theta(e') \mid w : e' \rightarrow e \}.
\]

1. Actions act on the objects of \( \text{Tpc}(a) \), to describe iteration of an action \( w : e \rightarrow e \) requires that \( e \) has states which is a level below objects.

2. \( W \) for workflow; in other contexts \( W(a) \) can be interpreted as routings, processes.
This gives $L(w_2 \circ w_2) = L(w_2) + L(w_1)$ as if $e$, $e'$ and $e''$ are chosen to maximize $L$ and $w_1 : e'' \to e'$ and $w_2 : e' \to e$ then $\theta(e) - \theta(e') + (\theta(e') - \theta(e'')) = \theta(e) - \theta(e'').$ □

A combinatorial system in which the objects in $Tpc(a)$ are or have a state in an ordered set and most $w \in W(a)$ decrease the state in their domain while adding to the state in their range is a supply network if there is a "replenishment logic" that allows each topic to be used indefinitely. The lead-time function adds to a combinatorial system a measure of dynamics or performance. The logic of manufacturing and supply networks applies to metabolism and ecosystems. Anabolism combines biochemicals for a range of responses so an organism can respond to outside stimulus. These processes deplete stocks of biochemicals circulating in the body and nutrients and time are required for replenishment. In an ecosystem every organism depletes its nutrients and the populations are in equilibrium when replenishment just exceeds depletion.

Examples of combinatorial systems include

1. "Symbolic" combinatorial system. These are systems in which the domain of map or action, $w$, is a set of objects or types $\{e_1, e_2, ..., e_n\}$ so that we have equations $w(e_1, e_2, ..., e_n) = e$. A symbolic combinatorial system replaces $w$ with a "merge" $[e_1, e_2, ..., e_n] = e$. All actions become "merges" making it explicit the way objects are "absorbed" in each $a \in \text{Obj}(A)$. Clearly all combinatorial systems have symbolic version. Symbolic combinatorial systems are useful in creating special examples. One can create mathematically legitimate constructions that need not have counterpart in the real world.

2. Linked Turing machines, or effective definitional schemes [28], with each $Tpc(a)$ the definition of a recursive function and each $Tpc(b)$ with $b \in S(a)$ processing data to input to $a$. Thus $b$ is an oracle [7] for $Tpc(a)$. For each $a Tpc(a)$ can be a different "signature" [32]. Large-scale systems are Turing machines with a very large number of oracles which parametrize responses of the system according to inputs.

3. Construction projects: the final step is $a^*$. The $Tpc(a)$ being the sub-projects activities.

4. Manufacturing and supply networks for consumer items with components drawn from many sources. $Tpc(a)$ corresponds the factories or work centers supplying components.

5. Combinatorial Modules. For the sub-category $\text{Comb}'$ of combinatorial systems for which maps in $\text{Map}(Tpc(a))$ do not form loops ("loop-free" systems) the subcategory of combinatorial modules is a retract. This will be proved when morphisms of combinatorial systems are introduced.

6. Multiple-scale systems. Here the states of the system at the largest scale can only be described in terms of an aggregation of data from smaller scales. The aggregation reduces the variation of the data or the dimensions of its state space. Such systems are illustrated by a simple example. Let $X = \{(x, y) \in \mathbb{N} \times \mathbb{N} | 0 < x, y \leq N\}$ for some $N$. $z \in X$ has states in an interval $St \subset \mathbb{Z}$. Each $z = (x_0, y_0)$ has a nearest neighborhood $N(x_0, y_0) = \{(x, y) \in X | x = x_0 \pm 1, y = y_0 \pm 1\} \cap X$ (the nearest neighbors truncated on
the boundaries). \(N^{k+1}(z_0) = \bigcup_{z \in N^k(z_0)} N(z)\). \(St(N^k(z)) = \{q : N^k(z) \rightarrow St\}\).

When there are no constraints this is \(\times St(z')\). Usually \(St(N^k(z))\) is "smoothed" in comparison to its constituent \(St(N^{k-1}(z))\). Functions \(f_i : St \rightarrow St\) are required to satisfy \(f(q(z)) = f(q(N(z)))\) and so are defined in terms of the state of the entire nearest neighborhood. Such functions loose information when iterated. The associated combinatorial system is as follows

(a) \(A = \{a_1 \prec a_2 \prec ... \prec a_n\}\),

(b) \(Obj(Tpc(a_1)) = \{(z, q) \mid z \in X, q \in St(z)\}\). \(Tpc(a_1)\) is a discrete category with the only maps being identities.

(c) \(Obj(Tpc(a_k)) = \{((N^k(z), q) \mid z \in X, q \in St(N^k(z))\}\) and \(Map(Tpc(a_k))\) are functions \(f_i : St(N^k) \rightarrow St(N^k)\) that satisfy the condition that

\[ f_i(q(z)) = f_i(q(N^k-1(z))) \]

so that \(f_i\) is determined at each point by the value of the states over the \(k-1\) nearest neighborhoods of elements in \(X\) that intersect \(N^k(z)\). This makes \(S(a_k) = a_{k-1}\) and for \(q \in St(N^k(z)), \beta(a_k, a_{k-1})(q) \in St(N^k-1(z))\).

Typically the nearest neighborhoods \(N(z)\) are not all the same. As \(X\) is finite there is a limit \(n\) so that \(N^n(z) = X\) unless \(X\) is in disjoint components.

More generally the states can be set of properties so that the \(P(f(z)) = f(P(N(z))\) where \(P\) is a property state.

In all of these cases there is a sense of progressing towards a final result. Something is being defined, be it a product, an artifact or information by combining other things. Although a combinatorial system can be seen as formal systems or definitional schema the added element of type weightings or lead-times give them concept of progress and dynamics which is the emphasis in this paper. The other aspect that is emphasized here is the combinatorial sub-modules which give a sense of "local" versus "entire" or "global".

### 2.1 Morphisms

A morphism \((A_1, Tpc_1 : A_1 \rightarrow Cat)) \rightarrow (A_2, Tpc_2 : A_2 \rightarrow Cat)\) is an order preserving function \(f : A_1 \rightarrow A_2\) and a family of functors \(Tpc(f) : Cat \rightarrow Cat\) such that

1. \(b \in S(a) \in A_1\) implies that if \(f(b) \neq f(a)\) then \(f(b) \in S(f(a))\) in \(A_2\),

2. for each \(b \in A_2\) with \(f^{-1}(b) = \{a_1, a_2, ... a_n\}\) for some \(n > 1\) then \(f^{-1}(b)\) is a combinatorial subsystem,

3. the following diagram of functors commutes

\[
\begin{array}{ccc}
A_1 & \xrightarrow{f} & A_2 \\
\downarrow{Tpc_1} & & \downarrow{Tpc_2} \\
Cat & \xrightarrow{Tpc(f)} & Cat.
\end{array}
\]
The family of functors \( Tpc(f) \) is indexed by \( A_1 \) and so are written as \( Tpc(f)(a) : Tpc_1(a) \to Tpc_2(f(a)) \).

The first condition implies that if \( \beta(a,b) \) is defined in \( A_1 \) and \( f(b) \neq f(a) \) implies \( \beta(a,b) \) has an image in \( A_2 \) and is therefore part of the category structure \( Tpc_2 \). This property of \( f \) means that the image of \( A_1 \) in \( A_2 \) is a combinatorial subsystem.

**Lemma 2.1.** If \( f \) is combinatorial system morphism then \( \forall a \in A_1, \, \theta(a)(e_2) > \theta(a)(e_1) \Rightarrow \theta(Tpc(f)(e_1)) \geq \theta(Tpc(f)(e_2)) \).

**Proof.** \( f : A_1 \to A_2 \) together with \( Tpc(f) \) gives \( Tpc(f) : Obj(Tpc_1(a)) \to Obj(Tpc_2(f(a))) \).

If \( e_1 \overset{w}{\to} e_2 \) in \( Tpc_1(a) \) then

\[
\begin{array}{ccc}
a_1 & \xrightarrow{w} & a_2 \\
\downarrow Tpc(f)(a) & & \downarrow Tpc(f)(a) \\
Tpc_2(f)(e_1) & \xrightarrow{Tpc(w)} & Tpc_2(f)(e_2).
\end{array}
\]

Which means that \( \theta(a)(e_2) > \theta(a)(e_1) \) implies \( \theta(f(a))TYP(f(e_2)) \geq \theta(f(a))TYP(f(e_1)) \).

**Definition.** A morphism \( f \) is a logistic morphism if the following diagram commutes.

\[
\begin{array}{ccc}
Tpc(f^{-1}(b)) & \xrightarrow{Tpc(f)} & Tpc(b) \\
\downarrow L & & \downarrow L \\
N & \xrightarrow{\geq} & N.
\end{array}
\]

That is lead-time through \( b \) is no greater than lead-time through \( f^{-1}(b) \).

Clearly combinatorial systems form a category \( Comb \). A simple proof shows that combinatorial systems with logistic morphisms form a subcategory of \( Comb \).

**Proposition 2.2.** If \( CombMod \) is the category of combinatorial modules it is a retract of \( Comb \).

**Proof.** (Sketch). Let \( Mod : Comb \to CombMod \) be the construction defined in the examples of combinatorial systems. If \( f : A_1 \to A_2 \) is a morphism in \( Comb \), \( Tpc(f)(a) : Obj(Tpc_1(a)) \to Obj(Tpc_2(f(a))) \) takes the basis of \( TYP_1(a) \) to that of \( TYP_2(f(a)) \) and so generates a homomorphism \( TYP(f) : TYP(a) \to TYP(f(a)) \). \( Tpc(f) \) being a functor means for each \( a \in A \) and \( e_1 \overset{w}{\to} e_2 \) we have \( (Tpc(f)(e_1)) \xrightarrow{Tpc(f)(w)} Tpc(f)(e_2)) \in Tpc_2(f(a)) \). Necessarily this preserves composition. Thus

\[
act(Tpc(a)) \xrightarrow{Tpc(f)} act(Tpc(f(a))
\]

preserves composition and so provides a homomorphism \( W(f) : W(a) \to W(f(a)) \).

This gives a homomorphism from action \( W(a) \times TYP(a) \xrightarrow{\alpha(a)} TYP(a) \) to the action \( W(f(a)) \times TYP(f(a)) \xrightarrow{\alpha(f(a))} TYP(f(a)) \).

Unless otherwise stated from now on all morphisms will be logistic.

If \( e \) is an object of \( Tpc(a) \) a stream to \( e \) is a set

\[
e_n \overset{w_n}{\to} e_{n-1} \overset{w_{n-1}}{\to} e_{n-2} \overset{w_{n-2}}{\to} \ldots \overset{w_1}{\to} e
\]
A stream can extend beyond $T_{pc}(a)$ into $S^n(a)$.

A stream is full if none of the included maps is a composition.

Let $T$ be a set of intervals in $\mathbb{N}$. $T$ is a set of time slots. For time slots $t_1$, $t_2$ write $t_1 > t_2$ if the least element of $t_1$ is not less than the greatest element of $t_2$. Let $\xi \subseteq Map(U) = \bigcup_{a \in U} Map(T_{pc}(a))$ be a set of full streams that can intersect.

**Definition** $\xi$ is a schedule if there is a function $s : \xi \to T$ such that

1. If $e'' \xrightarrow{w'} e' \xrightarrow{w} e$ is part of a stream in $\xi$ then $s(w) > s(w')$
2. Furthermore $s(w)$ is an interval $[n, n + L(w)]$ in $s(\xi)$.
3. If $\{a_1, a_2, a_3\} \subseteq U$ and $a_1 \in S(a_2)$ and $a_2 \in S(a_3)$ with $\xi \cap T_{pc}(a_3) \neq \emptyset$ and $\xi \cap T_{pc}(a_2) \neq \emptyset$ then $\xi \cap T_{pc}(a_1) \neq \emptyset$.

A schedule can be thought of as a network of maps flowing over part of $A$. If $s(w_1) \cap s(w_2)$ has more than two elements then $w_1$ and $w_2$ can occur at the same time. Even if $dom(w_2) \cap range(w_1) \neq \emptyset$ they can be in different streams.

**Definition** A cover $U$ of $A$ is a set of combinatorial subsystems in which every element of $A$ is in at least one element of $U$ and $U_1$ and $U_2$ in $U$ and $U_1 \cap U_2 \neq \emptyset$ then $U_1 \cap U_2 \in U$.

Let $Sch(U)$ be the set of schedules definable on $U \in U$. If $V \subseteq U$ there is clearly a restriction of $Sch(U) \to Sch(V)$ so we can define a presheaf $Sch$ of schedules.

**Proposition 2.3**. Sch is a sheaf.

**Proof.** Let $\xi_i$ be defined on $U_i$ such that $\xi_i = \xi_j$ on $U_i \cap U_j$ (the 0-cocycle condition) then as each is well defined on $U_i \setminus (U_i \cap U_j)$ we can define a $\xi_i$ on $U_1 \cup U_2$ that restricts to $\xi_i$ and $\xi_j$. In this way a schedule $\xi$ can be defined on $U = \bigcup_i U_i$ such that $\xi \mid U_i = \xi_i$. \hfill \Box

In the following $W(a)$ is used instead of $Map(T_{pc}(a))$ as "\(\)" indicates the presence of independent maps (as actions) and is a useful shorthand. Hopefully this minor abuse of notation will not confuse.

**Proposition 2.4**. A schedule in $U$ is a polynomial of maps in $W(U) = \bigcup_{a \in U} W(a)$.

**Proof.** Each independent stream $\xi : e_n \xrightarrow{w_n} e_{n-1} \xrightarrow{w_{n-1}} e_{n-2} \xrightarrow{w_{n-2}} \ldots \xrightarrow{w} e$ contributes $w \circ \ldots \circ w_{n-1} \circ w_n$. If $\xi' : e'_m \xrightarrow{w'_m} \ldots \xrightarrow{w'} e_j$ is another stream that terminates at $e_j$ in $a$ then we have $w \circ \ldots \circ w_j \circ w_{j+2} \circ \ldots \circ w_n \circ w' \circ \ldots \circ w'_m)$. Distributivity will give two streams identified in the time slots. In this way all streams contribute to the polynomial associated with $\xi$. \hfill \Box

The polynomial does not give the schedule in terms the function $s : \xi \to T$. It gives only the structure of the actions and with some ambiguity of order. For example $w_3, w_2, w_1 + w_4 + w_5 + w_6, w_4 + w_5$ does not order in time slots $w_4$ and $w_5$. In such cases there can be a number of ways in which the schedules can be implemented.

Identifying schedules with their associated polynomials allows us to add them.

**Lemma 2.2**. There is a number $\gamma(a)$ such that a schedule can be written as a set of maps $\xi(a) : T \to W(a)^{\gamma(a)}$.  

9
Proof. Let \( s(\xi) = T(\xi) \) then each \( w_i \) in \( \xi \) has a value \( s(w_i) \) in \( T(\xi) \). For a time slot \( t \in T(\xi) \) there is a set \( \{ w_1, w_2, ..., w_{n(t)} \} \) that is \( s^{-1}(t) \). Then \( s^{-1} : T(\xi) \to W(a_1) \oplus W(a_2) \oplus ... \oplus W(a_{n(t)}) \) where some of the \( a_k \) will be the same. Separating out the actions for a given \( a \), let \( \gamma(a) \) be the maximum of the \( t(n) \) for \( T(\xi(a)) \). We then have \( \xi(a) : T(\xi(a)) \to W(a)\gamma(a) \). Thus we can write the schedule as a collection \( \xi(a) : T \to W(a)\gamma(a) \).

**Proposition 2.5.** \( Sch \) is a group valued sheaf.

Proof. Given \( \xi_1 \) and \( \xi_2 \) in \( Sch(a) \) then by the lemma we can write \( \xi_i(a_{ij}) : T(\xi_i) \to W(a_{ij})\gamma(a_{ij}) \) where the \( \Sigma \) signifies addition across the separate components of \( W(a_{ij})\gamma(a_{ij}) \). If \( \xi_i \) is defined for \( U_i \subset A \) this makes each \( \xi_i \) a map \( \xi_i : T(\xi_i) \times U_i \to W(U_i) \). Addition \( \xi_1 + \xi_2 \) is now defined as the map \( \xi_1 + \xi_2 : (T(\xi_1) \cup T(\xi_2)) \times (U_1 \cup U_2) \to W(U_1 \cup U_2) \).

For the subcategory of \( Comb \) of logistic morphism we want \( Sch \) to be a functor. We need to impose an additional criteria on logistic morphisms.

**Definition** A morphism \( f : A_1 \to A_2 \) is single-layered on a covering \( U \) of \( A \) if for every \( U \in U \) \( f^{-1}(f(U)) = U \). Clearly single layered morphisms form a subcategory of \( Comb \). The condition \( f^{-1}(f(U)) = U \) is not a serious constraint in applications. Most morphism arise by rescaling (lumping sub-systems of topics into a single topic) or finding equivalence classes of types and actions.

**Theorem 2.3.** \( Sch \) is a functor from \( Comb \) and single layered logistic morphisms to the category of Abelian groups, \( Ab \).

Proof. Writing \( Sch(A) \) for the sheaf of schedules on \( A \) and \( Sch(A) \) for the global sections then we already have defined the functor on objects.

For a logistic morphism \( f : A_1 \to A_2 \) and for every \( U \in U \) with \( f^{-1}(f(U)) = U \) and \( \xi \in Sch(A_1)(U) \) define \( Sch(f)(\xi) \) as

\[
\xi \subseteq Map(Tpc(U)) \xrightarrow{f} Tpc(f)(\xi) \subseteq Map(Tpc(f(U)))
\]

wherein \( Tpc(f)(\xi) = \text{defn} Sch(f)(\xi) \subseteq Map(Tpc(f(U))) \). As the total duration of \( \xi \) = range of \( s(\xi) \) is the same as \( Sch(f)(\xi) \) then the time through \( U \) is the same as for \( f(U) \). Consequently \( Sch(A_1) \xrightarrow{Sch(f)} Sch(A_2) \) is a well defined homomorphism.

**Definition** A combinatorial scheme is a pair \( (A, P) \) with \( A \in Comb \) and \( P \) a group-valued sub-sheaf of \( Sch(A) \).

**Definition** The combinatorial schema of \( A \) is the finitely complete and cocomplete Abelian category generated by the category of group valued sheaves of schedules. The morphisms of this category are embeddings and epimorphisms constructed from logistic morphisms that induce sheaf homomorphisms over \( A \). Denote this category as \( Comb(A) \).

Why should we bother about sub-sheaves of \( Sch(A) \) treating them as special? A group valued sub-sheaf of \( Sch(A) \) is usually group valued for some reason. Often because it defines a class of schedules with added properties. These properties are, in turn, properties to the basic structure of \( A \). Thus an object \( P \) of \( Sch(A) \) can
be interpreted as schedules defined in special circumstances or on a model of \( A \).
This is a model in terms of model theory and corresponds to an image of a logistic
morphism in \( Comb \) \([16, 19]\). In terms of applications these become implementations
or manifestations of sets of algorithms. Each \( \mathcal{P} \) is then a class of dynamics that
demonstrate the capability of the system \( A \) under special constraints or boundary
conditions (something made clear in applications discussed below). Thus \( \mathcal{P} \) becomes
a mode of calculation using the conditions on the actions, types and schedules that
characterize the sheaf. This finite dynamics is played out in simulations of social and
natural systems of many types. In this case each sheaf is a parameterization of the
dynamics of \( A \).

The example of ecosystems. We now have sufficient concepts to sketch the
application of these ideas to ecosystems so fulfilling an earlier promise. Let \( A \) be the
species of an ecosystem. Each topic \( Tpc(a) \) is the life cycle of a species. \( Obj(Tpc(a)) \)
is the set of life stages of the species. For mammals these might be: dependent on
mother, independent juvenile, breeder, non-breeding adult; for reptiles: egg, hatch-
ing, juvenile, breeding adult. \( a_1 \in S(a_2) \) means some or all stages of \( a_2 \) use some
stages of \( a_1 \). If \( \beta(a_2, a_1) = Obj(Tpc(a_1)) \) then maps of \( Tpc(a_2) \) have the form
\( w_{a_2} : e_i \to e_{i+1} \) where \( e_i \) are stages of species \( a_2 \); \( \varepsilon \subseteq \beta(a_2, a_1) \) parameterizes the
movements from one stage of life of \( a_2 \) to the next. Maps have little content other
than being conditions for growing through a life stage. They are identified with the
numbers of required populations of \( S(a) \) being in sufficient or, for the species predia-
tors, limited numbers. If \( \| e \| \) is a measure of biomass then \( \| \beta(a_2, a_1)(e_i) \| \) gives
the amount of \( a_2 \) biomass need to sustain the biomass of \( e_i \). Some primary inputs
are not species but water, minerals, seed dispersants, shelter and area available to
a species. These can be external parameters the system. Thus the system \( A \) is a
trophic network perhaps parameterized by primary resources. The dynamics of the
system are in the schedules. Here a time interval must be chosen, probably by season,
but for some complex ecosystems of microorganisms (such as a compost heap) the
time interval might be in hours or minutes. A schedule \( \xi \) looks at the rate of “sup-
plies” being used by parts of the ecosystem. Lead times for maps hold the dynamics;
they are the growth rate of organisms. Where schedules match species replenish each
other - the necessary condition for stability. Where they do not and replenishment
fails some ecosystem change can be expected. This is different from dynamic systems
models of ecosystems where the dynamics is described by the trajectories of popu-
lation dynamics in the phase space of populations \([21, 24]\)\(^3\). This construction is an
investigation of how the ecosystem, seen as a supply network, must replenish itself to
move through the generations. Different sheaves of schedules can be parameterized by
different combination of primary resources, ecosystem biogeography and anticipated
or planned population statistics.

3 Defining Classes of Descriptions through Sheaves.

For a given combinatorial system \( A \) its combinatorial schema, \( Comb(A) \), will be the
main tool to analyze \( A \). Although \( Comb(A) \) contains all the information carried by
sub-sheaves of \( Sch(A) \) it gives no clue as to the way schedules might be distinguished

\(^3\)The combinatorial system approach presents the logic of the system with a different perspective.
It provides information for the interaction coefficients of the dynamic models which then is a test
for the combinatorial system model
by properties that are not formulated within the algebra of $A$. This section provides a general framework that uses the category of set valued sheaves defined on $A$ to define properties and classes of schedules. The advantage of this approach will be the definition of properties of the sheaves of $Comb(A)$ and not just properties of schedules or parts of schedules with the possibility that other parts have contrary properties.

3.1 Defining the sheaf of classes relative to a sheaf $K$

Let $U$ be a fixed covering of $A$ and let $Sh(A)$ denote the category of set valued sheaves of $A$ with this covering. Let $K \in Sh(A)$, a sheaf relation between $P \in Comb(A)$ and $K$ is a sheaf $\mathcal{L} \in Sh(A)$ with fixed maps $P \xrightarrow{proj_1} \mathcal{L}(P) \xrightarrow{proj_2} K$. This is to be a functor

$$[\mathcal{L}, K] : Comb(A) \to Sh(\mathcal{L}, K)$$

The category $Sh(\mathcal{L}, K)$ has objects pairs of maps $P \xrightarrow{proj_1} \mathcal{L}(P) \xrightarrow{proj_2} K$ and maps are commuting diagrams

$$P_1 \xrightarrow{proj_1} \mathcal{L}(P_1) \xrightarrow{proj_2} K$$
$$\downarrow P(f) \quad \quad \downarrow \mathcal{L}(f) \quad \quad \downarrow K(f)$$
$$P_2 \xleftarrow{proj_1} \mathcal{L}(P_2) \xleftarrow{proj_2} K$$

Definition: In $Sh(A)$ a $\mathcal{L}$-compatible map is a either a map $\bar{x} : K \to P$ such that $\bar{x} \circ proj_2 = proj_1$ or a map $\bar{y} : P \to K$ such that $\bar{y} \circ proj_1 = proj_2$.

Definition: The sheaves $\mathcal{P}_2$ and $K_2$, respectively, are defined as the colimits of $\mathcal{L}$-compatible maps in $Sh(A)$.

Proposition 3.1. The sheaf $\mathcal{P}_2$ is given as follows.

1. For $\kappa \in K(U)$, $X(\kappa) =_{def} \{ \xi \in P(U) \mid (\xi, \kappa) \in \mathcal{L}(U) \}$

2. $\mathcal{P}_2(U) =_{def} \{ X(\kappa) \mid \exists \xi \in P(U) \land (\xi, \kappa) \in \mathcal{L}(U) \}$ (that is $\kappa$ is in the image of $proj_2$).

Proof. The colimits exist in $Sh(A)$ as this category is an elementary topos and therefore complete and cocomplete. The colimit takes each element of $P(U)$ to the class defined by all maps with a common origin in $K$. That is, all the $\xi_1$ and $\xi_2$ that are joined by relation of $\mathcal{L}$ compatible maps $\bar{x}_1, \bar{x}_2 : K \to P$ wherein for each $U \in U$, $\bar{x}_i : \kappa \in K(U) \mapsto \xi_i$, $i = 1, 2$. This makes sets $\{ \xi \in P(U) \mid \exists \kappa \mathcal{L}(U)(\xi, \kappa) \in \mathcal{L}(U) \}$ the elements of $\mathcal{P}_2(U)$. Sheaf maps must preserve the restrictions and patching of sections. Thus for a set of $U_i$ with $U_i \cap U_j \neq \emptyset$ for pairs of $i$ and $j$ and $K(U_i) \xrightarrow{\bar{x}_i} P(U_i)$ with $\bar{x}_i \mid (U_i \cap U_j) = \bar{x}_j \mid (U_j \cap U_i)$ the $\bar{x}_i$ as sheaf maps must preserve the overlaps of sections. Thus for $\kappa_i$ defined on $U_i$, equal on the pairs of intersections and therefore having an extension on $U = \cup U_i$, the $\bar{x}_i(\kappa_i) = \bar{x}_j(\kappa_j)$ are equal on $U_i \cap U_j$ which gives a set $\xi_i = \bar{x}_i(\kappa_i)$ on the $U_i$ that patch together to make single section $\xi \in P(U)$. Thus we can define the sheaf $\mathcal{P}_2$ as stated.

$X(\kappa)$ can be identified as a property of schedules that can be formulated with reference to the section $\kappa$ of $K$. $\mathcal{P}_2$ is then the set of all $\mathcal{L}$ compatible classes of the schedules in $P$ that can be indexed in terms of $K$.

By the same reasoning we also have
Proposition 3.2. The sheaf $K_\Sigma$ is given as follows.

1. For $\xi \in \mathcal{P}(U)$ $Y(\xi) = \{ \kappa \in K(U) \mid (\xi, \kappa) \in \Sigma \}$

2. $K_\Sigma(U) = \{ Y(\xi) \mid \exists \kappa \in K(U) \wedge (\xi, \kappa) \in \Sigma(U) \}$ (that is $\xi$ is in the image of $\text{proj}_1$).

Corollary 3.1. The pairing $[\Sigma, K]$ gives a functor $(\_)_\Sigma : \text{Combi}(\Delta) \to \text{Sh}(\Delta)$ with $\mathcal{P} \mapsto \mathcal{P}_\Sigma$.

$Y(\xi)$ can be identified as the $\kappa$ in the sheaf $K$ that apply to at least one schedule in $\mathcal{P}$ satisfying the relation defined by $\Sigma$. That is $Y(\xi)$ are all the $\kappa$ properties that are true of $\xi$. Each element of $K_\Sigma$ is then a fiber over $\xi$ of the projection of $\Sigma$ as is each element of $\mathcal{P}_\Sigma$ a fiber over $\kappa$ of the projection of $\Sigma$. If we regard $K_\Sigma$ as a measure of relevance of $K$ for the properties of schedules we can ask how well it can discriminate among schedules and, in the case of each $\mathcal{P}$, how well $K$ discriminates among the schedules of $\mathcal{P}$. This discrimination would fail if for example $\mathcal{P}_\Sigma$ was always the same.

As a test of the discriminating power of the relation $\Sigma$ we can ask whether there are natural maps $f : \mathcal{P}_\Sigma \to \mathcal{K}_\Sigma$ such that $j \circ f = 1_{\mathcal{P}_\Sigma}$ and $j \circ f = 1_{\mathcal{K}_\Sigma}$. If $\mathcal{P}_\Sigma \to \mathcal{K}_\Sigma \to \mathcal{P}_\Sigma = 1_{\mathcal{P}_\Sigma}$ and $\mathcal{K}_\Sigma \to \mathcal{P}_\Sigma = 1_{\mathcal{K}_\Sigma}$ then $j$ and $f$ are bijective. This means that if $\xi_1$ and $\xi_2$ can be separated by any $\Sigma$ map $\mathcal{P}' \to \mathcal{P}$ they can be in different classes in $\mathcal{P}_\Sigma$. One of the clearest ways in which schedules can be distinguished is by size $\xi_1 \subset \xi_2$ or (equally) there is a non-zero difference in the associated polynomials. Most maps of sheaves that are induced by a logistic morphisms of $\Delta$ will maintain difference in size. Conversely if $\Sigma$ compatible maps maintain the distinction $\xi_1 \subset \xi_2$ then whatever relation $\Sigma$ creates with the elements of $\mathcal{K}$ that property is equally distinctive in $K$. This means the elements in $K$ and so the classes in $K_\Sigma$ are as equally divided as they are in $\mathcal{P}_\Sigma$.

Suppose now that a partial ordering can be imposed on sections of $K$, the indexing of elements in $\mathcal{P}_\Sigma$ define $\Sigma$ as order preserving if:

1. $(\xi_1, \kappa) \in \Sigma(U)$ and $\xi_2 > \xi_1$ then $(\xi_2, \kappa) \in \Sigma(U)$.

2. $(\xi, \kappa_1) \in \Sigma(U)$ and $\kappa_1 > \kappa_2$ then $(\xi, \kappa_2) \in \Sigma(U)$.

Lemma 3.2. $\Sigma$ is order preserving then

1. $\kappa_1 < \kappa_2$ implies $X(\kappa_2) \subseteq X(\kappa_1)$.

2. $\xi_1 < \xi_2$ then $Y(\xi_1) \subseteq Y(\xi_2)$.

This follows easily from the definitions.

Theorem 3.3. For $\Sigma$ order preserving, if for each $U \in \mathcal{U}$ the partial orders on $\mathcal{P}_\Sigma(U)$ and $K_\Sigma(U)$ are complete then there exist maps $j : \mathcal{P}_\Sigma \to \mathcal{K}_\Sigma$ and $j : \mathcal{K}_\Sigma \to \mathcal{P}_\Sigma$ such that $f \circ j = 1_{\mathcal{P}_\Sigma}$ and $j \circ f = 1_{\mathcal{K}_\Sigma}$.

Proof. Define $j : \mathcal{P}_\Sigma \to \mathcal{K}_\Sigma$ by $j(X(\kappa)) = \bigcap_{\xi \in X(\kappa)} Y(\xi)$. For this to be well defined it should correspond to a unique $Y(\xi)$. By the complete partial order condition $\xi$ exists. For each $\kappa$ in $Y(\xi)$ $\Sigma(\xi, \kappa)$ is true and is true for all $\kappa' \leq \kappa$. To pick out a $\kappa$ that characterizes $\xi$ we want to define $f : \mathcal{K}_\Sigma \to \mathcal{P}_\Sigma$ by $f(Y(\xi)) = \bigcap_{\kappa \in Y(\xi)} X(\kappa)$. Again as
the order in $\mathcal{P}_2(U)$ is complete there is a $\bar{\kappa}$ so that $f(Y(\xi)) = \bigcap_{\kappa \in Y(\xi)} X(\kappa) = X(\bar{\kappa})$. $\bar{\kappa}$ is the largest $\kappa$ for which $\mathfrak{L}(\xi, \kappa)$ is true for the given $\xi$.

As $\mathfrak{L}$ is order preserving, Lemma 3.1 gives:

1. $\kappa_1 < \kappa_2$ implies $X(\kappa_2) \subseteq X(\kappa_1)$ and $j(X(\kappa_1)) \subseteq j(X(\kappa_2))$.
2. $\xi_1 < \xi_2$ then $Y(\xi_1) \subseteq Y(\xi_2)$ and so $f(Y(\xi_2)) \subseteq f(Y(\xi_2))$.

Consequently $j$ and $f$ are order reversing.

Composing these maps, firstly,

$$X(\kappa) \xrightarrow{j} Y(\xi) \xrightarrow{\mathfrak{L}(\xi, \kappa')} \bigcap_{\xi, \kappa} X(\kappa') = X(\bar{\kappa}).$$

If $\xi \in X(\kappa)$ then $\xi$ belongs to all the $Y(\xi)$ determining $\xi$ so that $\xi$ is at least as big as $\xi$. Hence $\mathfrak{L}(\xi, \kappa') \Rightarrow \mathfrak{L}(\xi, \kappa)$ so $\xi$ is in the intersection defining $X(\bar{\kappa})$ so that $X(\kappa) \subseteq X(\bar{\kappa})$. If $\xi' \in X(\kappa)$ then, as $\mathfrak{L}(\xi', \bar{\kappa})$, $\mathfrak{L}(\xi', \kappa)$ is implied by $\mathfrak{L}(\xi', \kappa)$. But $\mathfrak{L}(\xi', \kappa)$ so $\mathfrak{L}(\xi', \kappa)$ and $\xi'$ is in $X(\kappa)$. Thus $f \circ j = 1_{\mathcal{P}_2}$.

Secondly,

$$Y(\xi) \xrightarrow{j} X(\bar{\kappa}) \xrightarrow{\mathfrak{L}(\xi', \kappa)} \bigcap_{\xi', \kappa} Y(\xi') = Y(\xi).$$

Suppose $\kappa' \in Y(\xi)$. By definition $\mathfrak{L}(\xi, \kappa)$. As $\kappa' < \kappa$ so $\mathfrak{L}(\xi, \kappa')$. But $\xi > \xi$ which implies $\mathfrak{L}(\xi, \kappa')$ and so $\kappa' \in Y(\xi')$ and therefore $Y(\xi) \subseteq Y(\xi)$.

If $\kappa'' \in Y(\xi)$ $\kappa''$ is in all the $Y(\xi')$ whose intersection gives $X(\bar{\kappa})$. In particular $\kappa'' \leq \kappa$ and as $\xi$ is the least $\xi$ for which $\mathfrak{L}(\xi', \kappa)$ and so $\mathfrak{L}(\xi, \kappa'')$ and $\kappa'' \in Y(\xi)$ and so $Y(\xi) \subseteq Y(\xi)$). This gives $Y(\xi) = Y(\xi)$ and so $j \circ f = 1_{\mathcal{P}_2}$.

As $j$ and $f$ are order reversing they give the required duality.

This theorem is an example of a class of theorems that establish the limits of discrimination of a language. In this sense it is an epistemological principle: for a given mode of analysis there are limits to discrimination of phenomena. The sheaf relation $\mathfrak{L}$ is an epistemological operator and handles the heterogeneous nature of the language of $\mathcal{A}$. The $[\mathfrak{L}, \mathcal{K}]$ giving a uniform approach to the logical relations arising from the connections of $\mathcal{A}$.

Not all languages implemented in sheaves $\mathcal{K}$ give rise to ordered classes. We can define a type protocol as a property that each $a \in A$ selects a set $E(a) \subseteq \text{Typ}(a)$ such that for $b \in S(a)$ and $E(a) \cap E(b) \subseteq \beta(a, b)$. We define a set valued sheaf $\mathcal{K}$ defined on the cover $\mathcal{U}$ of $\mathcal{A}$ such that $\mathcal{K}(U)$ is itself a cover for $E(U) = \bigcup_{a \in U} E(a)$. For $\kappa \in \mathcal{K}(U)$ $\mathfrak{L}(\xi, \kappa)(U)$ if and only if for all $w \in \text{range}(\xi), \text{dom}(w) \subseteq \kappa$. The sheaf nature of $\mathcal{K}$ imposes a connection among the $E(a)$ allowing the definition of protocol observant schedules to be a sub-sheaf of $\mathcal{P}$. Type protocols are found wherever special qualities of materials are required\(^4\). Similarly we can define action protocols.

\(^4\)Such as straight 'A' students, high-purity waters in medical compounds destined for immune compromised patients.
3.2 Dynamics: Measuring schedules

Definition A form on $Sch(A)$ is a measure on schedules with values in the real numbers.

Examples of forms are

1. $\Delta_{\theta} : Sch(a) \to \mathbb{N}$ gives the increase in type weighting per time slot so that $\xi \in Sch(A)$. $\Delta_{\theta}(\xi)$ can be taken as the per-time slots value added for $\xi$.

2. $a : Sch(a) \to \mathbb{N}$ the total number of actions of a schedule.

3. $L : Sch(a) \to \mathbb{N}$ the lead time of the schedule.

We shall assume that each form $\phi$ comes with an intended measure $m(\phi)$ which is an interval around some desired result. For $U \in \mathcal{U}$, $\phi$ acting on $\xi \in Sch(U)$, we expect $\phi(\xi) \in m(\phi)(U)$ meaning for $\forall a \in U$, $\phi(\xi(a)) \in m(\phi)(a)$. It is a filter for selecting schedules with a desired property.

Let $F(A) : \mathcal{U} \to \mathbb{R}$ be a selected ring of real-valued functions defined on each $U \in \mathcal{U}$. This is required to be a ring-valued sheaf.

Let $\Lambda$ be a finite set of forms

Definition The sheaf $\mathcal{D}[\Lambda]$ of forms of $\Lambda$ is the module of $F(A)$ forms generate by the forms $\lambda \in \Lambda$ so that for each $U \in \mathcal{U} \mathcal{D}[\Lambda](U)$ is the set of linear expressions $\phi(U) = \sum \lambda_i(U).\xi_i$ with $\lambda_i \in \Lambda$ and $\xi \in F(U)$ and with the restriction maps $\mathcal{D}[\Lambda](U) \to \mathcal{D}[\Lambda](V)$ given by $\phi(U)(\xi(U)) \mapsto \phi(V)(\xi | V) = \sum \lambda_i(V).\xi_i(V)$ whenever $V \subset U$ in $\mathcal{U}$.

As defined this is a presheaf, $\Lambda$ is chosen so that that $\mathcal{D}[\Lambda]$ is a sheaf. (We cannot always patch together forms such as the "width", the number of starting types.) The sheaf of forms for a specific combinatorial system, $\mathbb{A}$, is denoted $\mathcal{D}[\Lambda](\mathbb{A})$.

From hereon we assume that $\Lambda$ has been chosen and is fixed. As examples let $c : A \to \mathbb{R}$ (that is $c \in F(A)$) then two forms have immediate interpretation.

1. The "advantage" $\vartheta(a,c) = \Delta_{\theta} - c.a$, where $c(a)$ is a conversion from the units of the measure $a$ in each $a \in A$ to those of the weighting.

2. If costs are accrued per time $\vartheta(L,c) = \Delta_{\theta} - c.L$ is a related measure.

If $c(a)$ is cost per time, energy used, power consumption for each $w \in \xi(a)$, $\vartheta(L,c)(a)$ can measure the nett gain in type weighting (as a measure of value) for the time or energy required to produce the value while $\vartheta(a,c)(a)$ is the nett gain taking into the cost for each individual action. These forms can be "directed" to measure along a sub-combinatorial system $A' \subset A$ or modified to count or measure schedules restricted to classes of types.

Proposition 3.3. $\mathcal{D}$ is a covariant functor on $Comb$ with values in ring-valued sheaves.

Proof. As $Sch$ is a covariant functor, $f : A_1 \to A_2$ in $Comb$ such that $f$ maps $U_1 \to U_2$ implies $Sch(f) : Sch(A_1) \to Sch(A_2)$ maps schedules. Given $\phi \in \mathcal{D}(U)$ define $\mathcal{D}(f)$ by $\mathcal{D}(f)(\phi)(\xi) = \phi(Sch(f)(\xi)(f(U))) \in \mathcal{D}(A_2)(f(U))$. \qed

For a given $\mathcal{P}$ let $\mathcal{G} \to \mathcal{P} \times \mathcal{D}$ be given by $(\xi, \phi) \in \mathcal{G}(U)$ if and only if $\phi(a)(\xi) \in m(\phi)(a)$ is true for all $a \in U$. 

15
Lemma 3.4. $\mathfrak{g}$ is a sheaf.

Proof. Let $U_i \in U$, $U = \cup_i U_i$ and $\xi \mid U_i = \xi_i \in P(U_i)$, $\xi_i(U_i \cap U_j) = \xi_j(U_i \cap U_j)$ and $\phi_i \in D(U_i)$. From the fact that the $\xi_i$ are taken from a sheaf $P$ they will patch together to create a schedule $\xi$ defined in $U = \cup_i U_i$. The $\phi_i$ likewise patch together and constitute a form on $U$. (This follow from the product of two sheaves) and consequently we can defined $\phi(U)(\xi(U)) \in \mathfrak{g}$. □

(Alternatively $\mathfrak{g}$ could also be defined as a sub-sheaf. We can define the sub-presheaf as above and then take $\mathfrak{g}$ to be the associated sheaf. This avoids patching the $m(\phi)(U_i)$. The target measures $m(\phi)(a)$ are defined for each $a$ independently but have constraints that are necessary to defined $P_3$ below.)

3.2.1 The functors $P_3$ and $D_3$

We apply the results of the previous section with the substitutions $L \to \mathfrak{g}$, $K \to D$.

Define the sheaf $P_3$ as follows.

1. For $\phi \in D(U)$ $F(\phi) =_{defn} \{ \xi \in P(U) \mid \phi(\xi) \in m(\phi) \}$

2. $P_3(U) =_{defn} \{ F(\phi) \mid \phi \in D(U) \}$

Lemma 3.5. $P_3$ can be given a group operation.

Proof. For each $U \in U$ start with the free group generated by the set of classes $F(\phi) \in P_3(U)$ and factor by the sub-group of relations $F(\phi_1) + F(\phi_2) - F(\phi_3)$ for which $\xi \in F(\phi_1) \cap F(\phi_2) \Rightarrow \phi_1(\xi) + \phi_2(\xi) \in m(\phi_3)(U)$. That is as functions $\phi_1 + \phi_2$ defined on $F(\phi_1) \cap F(\phi_2)$ gives the same result on the same domain as $\phi_3$ to within the expected results on $P$. The form $0 : \xi \mapsto 0$ for all $\xi$ and which has $m(0) = \{0\}$ is the zero element. $\forall \xi \in P(U) \phi_1(\xi) + \phi_2(\xi) \in m(0) \Leftrightarrow F(\phi_1) + F(\phi_2) = F(0)$, $\phi_1(\xi) + \phi_2(\xi) \in m(0) = 0$ so $\phi_1 = -\phi_2$ on their target domains. □

The importance will be in equations $\sum_i F(\phi_i)(\xi) = \sum_j F(\phi_j)(\xi)$. If the $\phi_i$ deal with different aspects of a schedule, and this is where $\Lambda$ matters, these equations provide a linear equation representation of the relationship among measures. (This is important for the cohomology of $P_3$ as will be shown below.)

Definition. For $U \in U$, the sheaf $D_3$ is given by

1. For $\xi \in P(U)$, $\Delta(\xi) =_{defn} \{ \phi \in D(U) \mid \phi(\xi) \in m(\phi) \text{ in } U \}$. 

2. $D_3(U) =_{defn} \{ \Delta(\xi) \mid \xi \in P(U) \}$. 

$\Delta(\xi)$ can be identified as an abstract property associated with the set of measures of $\xi$ with values close to the target value of the measure.

Lemma 3.6. $D_3$ has a group structure.

Proof. For each $U \in U$ start with the free group generated by the set of classes $\Delta(\xi) \in D_3(U)$ and factor by the sub-group of relations given by $\Delta(\xi_1)(U) + \Delta(\xi_2)(U) - \Delta(\xi_3)(U)$ where for every $\phi_1 \in \Delta(\xi_1)$ and $\phi_2 \in \Delta(\xi_2)$ there is a $\phi' \in \Delta(\xi_3)$ such that $\phi_1(\xi_1) + \phi_2(\xi_2) \in m(\phi')$. If 0 is the "null schedule" for which all $\phi$ give the value zero, $\Delta(0)$ is the group identity. Thus if $\Delta(\xi_1) + \Delta(\xi_2) = \Delta(0)$ then for every pair $(\phi_1, \phi_2) \in \Delta(\xi_1) \times \Delta(\xi_2)$ there is a $\phi' \in \Delta(0)$ such that $\phi_1(\xi_1) + \phi_2(\xi_2) \in m(\phi')(0) = \{0\}$ making $\Delta(\xi_1)$ and $\Delta(\xi_2)$ inverse classes. □
3.3 Groupoid Fibrations as examples of the \([\mathcal{L}, \mathcal{K}]\) functors

Let \(\mathcal{P} \in Comb(\mathcal{A})\), a groupoid over \(\mathcal{P}\) [5] is a sheaf \(\mathcal{G}\) in \(Sh(\mathcal{A})\) with maps \(s, t : \mathcal{G} \rightarrow \mathcal{P}\) (the source and target of the groupoid action on a schedule) with the following properties.

1. There is an inverse isomorphism map \(t : \mathcal{G} \rightarrow \mathcal{G}\) such that \(s \circ t = t\) and \(t \circ t = s\).
2. \(g_1, g_2\) with \(s(g_1) = \xi_1\), \(t(g_1) = \xi_2\) and \(s(g_2) = \xi_2\), \(t(g_2) = \xi_3\) then \(g_2 \circ g_1\) is defined and \(s(g_2 \circ g_1) = s(g_1)\) and \(t(g_2 \circ g_1) = s(g_2)\).
3. \(s\) and \(t\) are onto and there exists a unit element for which \(s = t\).
4. \(s\) and \(t\) preserved associativity of \(\circ\)

This gives functor \([\mathcal{G}, 1_{Comb(\mathcal{A})}] : Comb(\mathcal{A}) \rightarrow Sh(\mathcal{A})[\mathcal{G}, 1_{Comb(\mathcal{A})}]\) with

\[
\mathcal{P} \mapsto (\mathcal{P} \overset{\xi}{\leftarrow} \mathcal{G} \overset{t}{\rightarrow} \mathcal{P}).
\]

This can be put in the form \(\mathcal{L}(\mathcal{P}) \mapsto \mathcal{P} \times \mathcal{G}\) with [\(\mathcal{L}, \mathcal{G}\)] given by

\[
\mathcal{P} \mapsto (\mathcal{P} \overset{\xi}{\leftarrow} \mathcal{G} \overset{t}{\rightarrow} \mathcal{P}),
\]

so that \(\text{proj}_1\) is identified with \(s\). Define \(\eta\) as

\[
\eta : (\mathcal{P} \overset{\xi}{\leftarrow} \mathcal{G} \overset{t}{\rightarrow} \mathcal{P}) \mapsto \mathcal{P} \overset{s}{\leftarrow} \mathcal{G} \overset{(s, 1)}{\rightarrow} \mathcal{L} \overset{t \circ (\text{proj}_2)}{\rightarrow} \mathcal{P}.
\]

This gives a functor \(\eta\) which has an obvious inverse.

\[
\begin{array}{ccc}
Sh[\mathcal{L}, \mathcal{G}] & \overset{\eta}{\longrightarrow} & Sh[\mathcal{G}, 1_{Comb(\mathcal{A})}] \\
\end{array}
\]

We now have sheaves \(\mathcal{P}_\mathcal{G}, \mathcal{G}_\mathcal{G}\) given by:

1. \(X(g) = \{\xi \in \mathcal{P}(U) \mid sg = \xi\}\) and so
2. \(\mathcal{P}_\mathcal{G}(U) = \{X(g) \mid g \in \mathcal{G}(U) \land (sg = \xi) \land (g \neq 1)\}\). This is the set of \(\mathcal{P}\) that have non-trivial groupoid actions.
3. \(Y(\xi) = \{g \in \mathcal{G}(U) \mid (g \neq 1) \land (sg = \xi)\}\) and
4. \(\mathcal{G}_\mathcal{G}(U) = \{Y(\xi) \mid \xi \in \mathcal{P}(U)\}\).

\(Y(\xi)\) is the set of \(g\) that start at \(\xi\) and hence can be mapped to the set of \(\xi'\) for which \(g : \xi \rightarrow \xi'\). In Part 2 we shall define this as a section

1. \(G(U)(\xi) = \{\xi' \in \mathcal{P}(U) \mid \exists g \in \mathcal{G}(U)(g \neq 1) \land (sg = \xi) \land (tg = \xi')\}\) and so
2. \(\mathcal{G}_\mathcal{G}(U) = \{G(U)(\xi) \mid \xi \in \mathcal{P}(U)\}\).

which is equivalent to the definition above. Properties of these functors will be discussed in Part 2.


4 Cohomology

The \([\mathcal{E}, \mathcal{K}]\) constructions point to classes of derived structures of a sheaf \(\mathcal{P}\). In the applications we shall find that important information about a sheaf is in the relations that hold among sets of schedules that share properties at least locally. The technique of choice to make this explicit is cohomology. Given a group valued sheaf \(\mathcal{P}\) we can define the Čech cohomology on \(\mathcal{P}\) giving the groups \(\check{H}^n(\mathcal{P})\). There are many references to this cohomology with [11, 30, 10] spanning its use in algebraic geometry, algebraic topology can analysis.

In the following we use the convention that if \(U_{i_j} \in \mathcal{U}\) and \(U_{i_0} \cap U_{i_1} \cap \ldots \cap U_{i_n}\) is non-empty then \(U[i_k] = \cap_{j \neq k} U_{i_j}\). As usual denote the cochains by

\[
C^n = C^n(\mathcal{P}, \mathcal{U}) = \prod_{i_0, i_1, \ldots, i_n} \mathcal{P}(U_{i_0} \cap U_{i_1} \cap \ldots \cap U_{i_n})
\]

where the product is over \(n+1\) tuples \(i_0 < i_1 < \ldots < i_n\) giving \(n\) intersections. Define the coboundary map \(\partial^n : C^n \to C^{n+1}\) by setting

\[
\partial^n(\xi)(U_{i_0} \cap U_{i_1} \cap \ldots \cap U_{i_n}) = \sum_{k=0}^{n+1} (-1)^k \xi_{i_k} | (U_{i_0} \cap U_{i_1} \cap \ldots \cap U_{i_n} \cap U_{i_{n+1}})
\]

where \(\xi = <\xi_{i_k} \in \mathcal{P}(U[i_k]), k = 0, 1, \ldots, n + 1\).

Notation:

1. The subgroup of \(n\)-cocycles in \(C^n, Z^n(\mathcal{P})\), is the group \(\ker(\partial^n)\). We write \(Z^n\) when \(\mathcal{P}\) is implied.

2. The subgroup of \(n\)-co-boundaries in \(C^n, B^n(\mathcal{P})\), is the group \(\im(\partial^{n-1})\). We write \(B^n\) when \(\mathcal{P}\) is implied.

Definition The group of \(n\) cohomology classes, or the \(n\)-th cohomology group is

\[
\check{H}^n(\mathcal{P}) = \ker(\partial^n)/\im(\partial^{n-1}) = Z^n/B^n.
\]

The 0-th cohomology class is defined by an augmentation map. We augment the cochain complex with the a map

\[
\varepsilon : \mathcal{P}(A) \to C^0(\mathcal{V}, \mathcal{P}) = \prod_i \mathcal{P}(P_i).
\]

The value of \(\varepsilon(\xi)\) is \(\xi|U\) in each \(\mathcal{P}(U)\). \(B^0\) is then taken to be defined as the image of \(\varepsilon\). \(\check{H}^0(\mathcal{P}) = Z^0/B^0\) is then seen to be the global sections which in this case are schedules defined over all of \(A\).

We shall presently show that \(\check{H}\), despite its name, is a covariant functor in this context. It is presented in terms of schedules so in terms of generalized flows but is, above all, a measure of locally defined linear relations beyond those of the schedules themselves. In the following we adopt the conventions

1. In an expression such as \(U_{i_0} \cap U_{i_1} \cap \ldots \cap U_{i_k} \cap \ldots U_{i_n}\), \(\cap\) denotes omission of the set, in this case \(U_{i_k}\).

2. A cocycle equation \(\xi_0 - \xi_1 + \ldots + (-1)^n \xi_n\) | \((U) = 0\) will usual be written as \((\xi_1 + \xi_2 + \ldots + \xi_{2k-1})\) | \(U = \xi_0 + \xi_2 + \ldots + \xi_{2k}\) | \(U\) or \(\sum_{k \text{ odd}} \xi_k = \sum_{k \text{ even}} \xi_k\) or a similar variation.
The following illustrates cohomology as indicative of equations among types and actions. With \( \xi_k \) being defined on \( U[k] = \cap U_i \) let the total time slots available in \( U = \cap U_i \) be the set \( T(U) = \{ t_1, t_2, ..., t_n \} \). On \( U \) the equation

\[
\sum_{k \text{ even}} \xi_k(t_i) = \sum_{k \text{ odd}} \xi_k(t_i); \quad t_i \in T(U)
\]  

holds. Let \( \xi_k(t_i) = \sum w_{k,j}(t_i) \) in the above equation giving

\[
\sum_{k \text{ even}} \sum w_{k,j}(t_i) = \sum_{k \text{ odd}} \sum w_{k,j}(t_i).
\]

Even in the simplest case of \( Z_0 \) with \( \xi_0 = \xi_1 \) on \( U = U_0 \cap U_1 \) these equations imply \( \sum w_{0,j}(t_i) = \sum w_{1,j}(t_i) \)). The equations in actions translate to equations in types. Suppose \( w_0(t)(e_1, e_2) = e \) and \( w_0 = w_{1,1} + w_{1,2} \) on \( U \) so that \( (w_{1,1} + w_{1,2})(e_1, e_2) = e \). This forces a decomposition of \( e = e' + e'' \) and corresponding breakup of \( (e_1, e_2) \) to \( e_i = \sum a_{i,l} e_{i,l} \). These equations can change from time slot to time slot. For a cocycle not to be a coboundary then at least one of these equations has to be specific to \( U \) and not in all of the \( U_k \).

Suppose we have \( \zeta_1 = \sum k \text{ odd} \xi_k = \sum k \text{ even} \xi_k = \zeta_2 \) on \( U \) and \( T(U) \) as above, then each \( t_i \) \( \zeta_j(t_i) : Typ(U) \rightarrow Typ(U), \) \( j = 1, 2, i = 1, 2, ..., n \). These can be written as matrices and the composition of the matrices can be written as \( M(\zeta_j) : I \rightarrow O, I \) being the starting types and \( O \) the output types of the \( M(\zeta_j) \). This is also true at each stage \( M(\zeta_j)(t_i) = I(i) \rightarrow I(i + 1) \) which provides the factorization of each \( M(\zeta_i) \).

**Proposition 4.1.** A cocycle exists if and only if there is a set of linear isomorphisms \( L(i) \) for each \( t_i \) in \( T(U) \) so that the following diagrams commute.

\[
\begin{array}{c}
I(i) \xrightarrow{M(\zeta_1(t_i))} I(i+1) \\
\downarrow \quad \quad \downarrow \quad \quad \downarrow \quad \quad \downarrow \quad \quad \downarrow \quad \quad \downarrow \\
L(i) \xrightarrow{M(\zeta_2(t_i))} I(i+1)
\end{array}
\]

**Proof.** Each \( w \) in \( \zeta_1(t_i) \) corresponds to a mapping \( w : \sum_{i} x_i e_i \mapsto \sum_{m} y m e'_m \) and \( M(\zeta_1(t_i)) \) is the sum of such transformations written in matrix form. For a cocycle to exist \( w \) must be part of a linear expression \( \sum_p w_p \in \zeta_1(t_i) \) that can be set equal to an expression \( \sum_q w_q \in \zeta_2(t_i) \). For these actions to have the same effect the combined effect of the \( w_p \in \zeta_1(t_i) \) in \( \sum_p w_p \) can be written

\[
\sum_{p} w_p : \sum_{p} x_p e_{p,t} \mapsto \sum_{p} y_{p,m} e'_{p,m},
\]

this matches the corresponding effect of the \( \sum_{q} w_q \) in \( \zeta_2(t_i) \) which gives

\[
\sum_{q} w_q : \sum_{q} x_{q,t} e_{q,t} \mapsto \sum_{q} y_{q,m} e'_{q,m}.
\]

This provides, after re-arrangement, linear expressions \( L(i) \) and \( L(i+1) \) matching the transformations of the types during the time slot \( t_i \). Thus the existence of the \( L(i) \) and \( L(i+1) \) transformations follows from the existence of a cocycle.

If these diagrams commute then at each stage in \( T(U) \), \( \zeta_1 \) and \( \zeta_2 \) produce the same effects and so the combined affects of the actions are the same. \( \square \)
Illustrations. Simplest case. 1 cocycle. $\text{Typ}(U) = \{e_0, e_2, e_3\}$ and $\xi_0$ in $U$ is $w_0(e_1, e_2) = e_0', \xi_2$ in $U$ is $w_2(e_0, e_1) = e_2', \xi_1$ in $U$ is $w_1(e_0, 2e_1, e_2) = e_1'$, then $e_1 = e_0 + e_2$ means $\xi_1 = \xi_0 + \xi_2$ in $U$. This becomes a matrix equation in the ring $\text{GL}(Z, 3)$ giving the possibility with links to other parts of algebra.

Example 1. $\xi_0$ in $U$ is $w_0(e_1, e_3) = e_0', \xi_2$ in $U$ is $w_2(e_1, e_3) = e_2', \xi_1$ in $U$ is $w_1(e_0, e_2) = e_3'$, then $e_1 + e_3 = e_0 + e_2$ and $e_1' + e_3' = e_0' + e_2'$ means $\xi_1 + \xi_3 = \xi_0 + \xi_2$ in $U$.

Example 2. $\xi_0$ in $U$ is $w_0(e_1, e_2, e_3) = e_0', \xi_2$ in $U$ is $w_2(e_1, e_3) = e_2', \xi_1$ in $U$ is $w_1(e_0, e_1, e_2) = e_3'$, then $2e_1 + e_2 + e_3 = 2e_0 + e_1 + e_2$ and $e_1' + e_3' = e_0' + e_2'$ means $\xi_1 + \xi_3 = \xi_0 + \xi_2$ in $U$. A solution is $2e_0 = e_1$ and $e_2 = e_3$.

Cohomology classes are created by cocycles that are not coboundaries. What does this condition mean?

If the restriction map $\mathcal{P}(U_1) \to \mathcal{P}(U_2)$ is an epimorphism you can always find schedules $\xi, \eta_1, \eta_2$ such that $\xi(U_2) = \eta_1 \mid U_2 - \eta_2 \mid U_2$ where the $\eta_i$ are defined on $U_1$. If $U = \cap_i U_i = \cap_k U[k]$ and the image of

\[
\oplus \mathcal{P}(U[k]) \xrightarrow{\oplus \rho(U[k], U)} \mathcal{P}(U)
\]

($\rho(U[k], U)$ the restriction map) is not onto then the image of $\partial$, so $B^n$, is also not all of $C^n$. This is a necessary condition for the existence of cohomology groups. This is typically the case when some of the $\xi \in \mathcal{P}(U)$ can create larger quantities of a type than can be absorbed or used in elements of $a \in U[k] \setminus U$ for which $U \cap \xi\infty(a)$ is not empty. In particular if $\xi \in \mathcal{P}(U)$ can produce beyond what $a$ can absorb then $\xi$ cannot be the difference of two schedules defined on $U[k]$ and consequently $\mathcal{P}(U[k])$ cannot generate $\mathcal{P}(U)$. This is true whether the limits on $a$ are in terms of quantities or rates of processing.

The most extreme case is that if $\mathcal{P}(U)$ is generated by a single schedule $\zeta$ which cannot be decomposed. Such schedules exist for monopolies, single source of expertise or key ideas. In certain circumstances these split the cohomology groups as shown below.

**Proposition 4.2.** Suppose every global schedule $\xi \in \mathcal{P}(A)$ has $\xi \mid U = \zeta$ and that $U$ is a "bottleneck": $S^{+\infty}(U) \cup S^{\infty}(U)) \cup U = A$. Then

\[
\check{H}^n(B) = \check{H}^n(\mathcal{P}_{S^{+\infty}(U)}) \oplus \check{H}^n(\mathcal{P}_{S^{\infty}(U)})
\]

where $\mathcal{P}_B$ is the restriction of the sheaf $\mathcal{P}$ to the combinatorial subsystem $B$.

**Proof.** $\zeta$ does not appear in any cocycle or coboundary equation either within $\mathcal{P}(U)$ or in the complement of $U$. Any equation involving a set of schedules $\xi_i$ defined on $U_i \supset U$ is restricted to the parts outside $U$. As $\xi_i \mid U = \zeta$ any equation $\Sigma \xi_i \mid U = \Sigma \xi'_i \mid U$ must restrict to an equation $n.\zeta = m.\zeta$ and so $m = n$ on $U$. This will always be a coboundary as $\zeta + \eta = \eta'$ restricted to $U$ becomes $\zeta + (n - 1).\zeta = n.\zeta$ for some $n$. In the complement of $U$ schedules can be separated into the parts in $S^{+\infty}(U)$ or $S^{\infty}(U)$ and will not contain $\zeta$. Furthermore the variation in schedules is again outside $U$. This means all equations among the schedules in $S^{+\infty}(U)$ are independent of those in $S^{\infty}(U)$ giving the cohomology group decomposition. □
If \( U = \{ a \} \) \( \zeta \) can arise because of a single action.

**Definition.** An action \( w_0 \in W(a) \) is a maximum capacity action if for all \( \xi \in \text{Sch}(a) \), \( w_0 \in \xi(t) \Rightarrow \forall w' \neq w_0 \ w' \) is not in \( \xi(t) \) and all complete schedules have to pass through (or use) \( w_0 \). The proof of proposition 4.2 can be localized to \( a \) with \( w_0 \) playing the role of \( \zeta \).

Less restrictive is the concept of a maximal schedule. Here a maximal schedule is one that uses all available slots. Another schedule cannot be added to it in the same duration. So \( \zeta : T \rightarrow W(U) \gamma(U) = \bigoplus_{a \in U} W(a) \gamma(a) \) is maximal if there is no non-negative schedule \( \xi' \in \text{Sch}(U) \) with \((\zeta + \xi') : T \rightarrow W(U) \gamma(U)\).

A maximal schedule allows decompositions so a maximal \( \zeta \) can equal \( \Sigma \xi_i \).

**Proposition 4.3.** If an element \( \zeta \) of \( \mathcal{P}(U) \) is a maximal element so that there are no elements \( \eta_1 \) and \( \eta_2 \) (that are positive) such that \( \zeta + \eta_1 = \eta_2 \) then for each decomposition of \( \zeta \) there is a cohomology class for \( \mathcal{P} \).

**Proof.** If \( \xi_i \), defined on \( U_i \), are such that \( \zeta - \Sigma \xi_i = 0 \) on \( \cap U_i \) then with appropriate signs changed the equation is a cocycle equation. \( \zeta \) cannot be represented as the sum of the differences therefore it cannot be coboundary. This means not all of the sections defining the cocycle can be a coboundary and hence the cocycle is a (non-trivial) cohomology class.

The choice of the cover \( \mathcal{U} \) for a combinatorial system \( \mathbf{A} \) has a significant effect on the cohomology as the following result shows.

**Theorem 4.1.** Let \( \mathcal{U} = \{ U_1, U_2, ..., U_N \} \) be a cover for \( \mathbf{A} \) such that there is number \( n_0 < N \) such that any set of \( n > n_0 \) intersections \( U_{i_0} \cap ... \cap U_{i_n} \) is equal to the intersections of a subset of those sets with no more than \( n_0 \) members. If \( \mathcal{P} \) is defined on \( \mathcal{U} \) then \( H^{n_0+k}(\mathcal{P}) = 0 \) for \( k \geq 1 \).

**Proof.** This follows as any cocycle on more than \( n > n_0 + 1 \) intersections is an equation on \( n - 1 \) intersections and hence can be represented as coboundaries. Specifically, let \( \langle \xi_j \rangle \) form a \( n \)-cocycle. \( \xi_j \) is defined on \( U_{i_0} \cap ... \cap U_{i_n} \). Hence \( \Sigma (-1)^j \xi_j = 0 \) on \( U_{i_0} \cap ... \cap U_{i_n} \). For \( n > n_0 + 1 \) we need to show this is a coboundary. If \( n > n_0 + 1 \) then for each \( j \) there is an \( l \) such that \( U_{i_0} \cap ... \cap U_{i_j} \cap ... \cap U_{i_n} = U_{i_0} \cap ... \cap U_{i_j} \cap ... \cap U_{i_l} \cap ... \cap U_{i_n} \) so \( \mathcal{P}(U_{i_0} \cap ... \cap U_{i_j} \cap ... \cap U_{i_n}) = \mathcal{P}(U_{i_0} \cap ... \cap U_{i_j} \cap ... \cap U_{i_l} \cap ... \cap U_{i_n}) \). This means each \( \xi_j \) can be written as an alternating sum of schedules defined on fewer intersections of the cover and hence can be expressed as a coboundary. Thus for \( n > n_0 \) \( H^n(\mathcal{P}) = 0 \).

For this reason we use a ”standard” cover: \( \mathcal{U} \) is the set of sets \( S^k(a) \) for all \( a \in A \) together with their intersections. This is a very redundant cover and many useful covers will have fewer sets.

### 4.1 Cohomology of Combinatorial Modules

This section demonstrates the variety of cohomologies of combinatorial modules. We adopt the language of supply networks and see schedules as flows of types.

\(^5\text{Although } \text{Sch} \text{ is a group and we can freely add schedules there are many relationships among the schedules so that } \text{Sch} \text{ is not merely a free group on every } U \in \mathcal{U}.\)
4.1.1 A linearly ordered combinatorial modules

A combinatorial system $\mathcal{A}$ is linearly ordered if $\mathcal{A}$ is a set

$$\{a^* = a_0 \succ a_1 \succ a_2 \succ \ldots a_{n-1} \succ a_n\}.$$ 

**Proposition 4.4.** Given a cohomology "specification" of $\dim H^p(\mathcal{P}) = m$ there exists a linearly ordered combinatorial module $\mathcal{A}$ and a $\mathcal{P} \in \text{Comb}(\mathcal{A})$ that satisfies this specification.

**Proof.** We construct $\mathcal{A}$ as follows: $a^*$ has one type $e_0$ and $\text{Typ}(a_i) = \{e_{i,1}, e_{i,2}, \ldots, e_{i,i}\}$ and every element $e_{i-1,r} \in \beta(a_{i-1}, a_i) \subset \text{Typ}(a_{i-1})$ is the result of actions that merge a set $E_{i-1,r} \subset \text{Type}(a_i)$ to $e_{i-1,r}$ (this illustrates the construction of a symbolic combinatorial system). Furthermore the sets $E_{i-1,r}$ cover $\text{Typ}(a_{i-1})$. Let $U$ be the set of intervals $U(i,j) = \{a_i \succ a_{i+1} \succ \ldots a_j\}$ with $j \geq i + 2$. (This can also be characterized as the sets $S^{\leq \infty}(a_i) \cap S^{+\leq \infty}(a_j)$ with $j \geq i + 2$).

We impose a protocol $P$ on the types of $\mathcal{A}$ such that $P(a_i) \subseteq 2^{\text{Typ}(a_i)}$. $U = \{a_{k'}, a_{k'+1}, \ldots, a_{k''}\}$ is chosen so that for $n$ sufficiently large this allows the selection of sets of distinct subsets $U_j \supseteq \cup_{j'} S^{+(k''-k')} + \alpha(a_{k''})$ such that $\cap_j U_j = U$. Select $\mathcal{P} \in \text{Comb}(\mathcal{A})$ that observes the protocol for schedules defined on $\{a_1, a_2, \ldots, a_{k''}\}$ but not in schedules defined on $\{a_{k'}, a_{k'+1}, \ldots, a_{k''}\}$. As $\mathcal{P}(U_j)$ observes the protocol the restriction map $\mathcal{P}(U_j) \rightarrow \mathcal{P}(U)$ is not onto.

Let the sets $U[l] = U_{j_0} \cap U_{j_1} \cap \ldots \cap U_{j_p}$ be selected, then $U = \cap U[l]$. Furthermore $\mathcal{P}(U[l]) \rightarrow \mathcal{P}(U)$ is not onto. Let schedules $\xi_l$ be defined on $U[l]$ so that we have a cocycle $\sum_{l \in \text{even}} \xi_l | U = \sum_{l \in \text{odd}} \xi_l | U$. If this cocycle is specific to $U$ and hence not in the image $\cup_{l \in \text{even}} \mathcal{P}(U[l]) \rightarrow \mathcal{P}(U)$ we can arrange that this arises because of the mixing of the sets $E_{k'+j,r}$, $j = 1, 2, \ldots$ via equations specific to the types and actions in $U$. This means that the cocycle involves a set of linear equations among the $E_{k'+j,r}$ and other possible $E_{k'+r}$ of $\text{Typ}(U)$. $\mathcal{A}$ is constructed so that:

1. Each such cover $E_{i,r} E_{i',r}, \ldots$ of $\text{Typ}(a_i)$ are linked by equations that give rise to a new cocycle.

2. $n$ is sufficiently large so that $p$ distinct sets of intervals $U^p[k]$ with $U$ as their intersections can be selected.

3. The dimension of $\text{Typ}(U)$ is sufficiently large so that there are $m$ distinct sets of sets $E_U \subset \text{Typ}(U)$ that are the types involved in the equations between schedules $\xi^p_m(k)$ defined on the $p$ different sets of $U^p[k]$.

Each cocycle not a coboundary then provides a dimension to $Z^p$ and so to $H^p(\mathcal{P})$. Indeed it is sufficient that we have $m$ such different "recipes" for any $e_i$ in the protocol set in $\text{Typ}(a_{k'+q'})$, $0 \leq q' \leq q$ where the recipes include types from outside the protocol.

In the case where the larger the $i$ the greater the capacity of $a_i$ and the faster it can produce then storage capabilities can be further manipulated to ensure the restriction map $\mathcal{P}(S^{\leq \infty}(a_i)) \rightarrow \mathcal{P}(S^{\leq \infty}(a_{i+1}))$ is not onto for all $i$. 

\[22\]
4.1.2 Constructing systems with arbitrary cohomology

This section outlines a general construction of combinatorial modules to have a "cohomological specification": \( \dim(H^p(\mathcal{P}) \geq d(p)) \), as given in the linear case in Proposition 4.5. We shall prove:

**Theorem 4.2.** Given an arbitrary set of numbers \( d(p) \in \mathbb{N} \) for \( p = 1, 2, \ldots, n \) there exists a combinatorial module \( \mathcal{A} \) with \( \mathcal{P} \in \text{Comb}(\mathcal{A}) \) such that \( \dim(H^p(\mathcal{P})) \geq d(p) \).

**Proof.** We construct \( \mathcal{A} \) as a series of separate systems and then join them all at a single \( a^* \). Let \( \mathcal{A}_i \) be given thus. There are \( l \) sets \( U_{i,i}, i = 1, 2, \ldots, l \) each containing at least 2 elements of \( \mathcal{A}_i \) and have \( S^+(U_{1,i}) = a_0 = a^*_i \). For \( i = 1, 2, \ldots, l \) \( S^{\leq k}(U_{1,i}) \subset V_i \) and \( (i) U_{1,i} \cap V_i \neq \emptyset \). That is \( V_k \) is the common source of types that flow to the \( U_{1,i} \).

In the diagram below the arrows denote the flow of the schedules.

\[
\begin{array}{ccc}
   a_0 = a^*_i & U_1 & U_2 \ldots \ldots \ldots U_l \\

   V_1 & U_l & \end{array}
\]

We assume \( Typ(a_0) = \{e_1, e_2, \ldots e_l\} \). Each \( U_{1,i} \) deals with all types except the type \( e_i \). Each \( U_i \) has a structure in which each \( e_i, i \neq l \) combines a set \( E_i \subset Typ(U_{1,i}) \). For each \( E_i, V_i \) contains a much larger set \( E_i \) which can serve as \( \beta(U_{1,i}, V)(E_i) \) (abusing notation). Each \( \mathcal{A}_i \) is equipped with a protocol and a sheaf \( \mathcal{P} \in \text{Comb}(\mathcal{A}_i) \) that is protocol invariant in each \( U_{1,i} \). Furthermore we can assume \( E_i \) is the protocol types for \( e_i \).

Thus \( \mathcal{A}_i \) has a non-empty intersection \( U_i = \bigcap V_i \cap U_{1,1} \cap U_{1,2} \cap \ldots \cap U_{1,l} \) so \( C^1(\mathcal{P}) \) contains the components \( \mathcal{P}(U_i) \). As there is no \( C^{d+1} \), \( Z^2(\mathcal{P}) = C^1(\mathcal{P}) \). The construction of \( \mathcal{A}_i \) needs to ensure \( B^i(\mathcal{P}) \neq C^1(\mathcal{P}) \) and indeed \( \dim(C^1(\mathcal{P})) \) can be made arbitrarily large.

Our construction means \( U[k] = (V_i \cap U_{1,1} \cap U_{1,2} \cap \ldots \cap U_{1,l}) \) is required for \( e_k \).

\( B^i(\mathcal{P}) = im(\partial^{-1}P(U[k]) \subseteq C^1(\mathcal{P}) \) is satisfied if there are \( \xi \in P(U) = C^1(\mathcal{P}) \) that are not in the restriction of any of the \( \xi_k \in P(U[k]) \). Such \( \xi_k \) cannot provide for \( e_k \).

Consequently the existence of many schedules in \( P(U) \) that act on all primary types destined for \( Typ(a_0) \) means that the restriction of \( P(U[k]) \subseteq P(U) \). Furthermore, we can assume there is a set \( P(V_i) = U_i \beta(U, V)(E_i) \subseteq Typ(V_i) \) that is the bill of types for the \( E_i \). Now if each type in \( Typ(U) \) has at least \( d(l) \) ways of being merged from starting types with linear maps among sets of \( e^c \) in \( Typ(U) \) we can easily create sufficient non-protocol schedules to ensure \( C^1(\mathcal{P}) \) has all \( d(l) \) independent schedules classes.

We create \( \mathcal{A} \) by joining all the \( \mathcal{A}_i \) together at \( a^* = \{a^*_1, a^*_2, \ldots, a^*_l\} \) with \( Typ(a^*) = \bigcup_{i=1}^l Typ(a^*_i) \). It is clear that \( 
\end{array}
\]

4.2 The Cohomology of \( \mathcal{P}_\delta \)

In this section we investigate \( H^n(\mathcal{P}_\delta) \). Although the groups \( H^n \) have been defined on group valued sub-sheaves of \( Sch \) they can be defined on any group valued presheaf defined for some appropriate cover of \( \mathcal{A} \). Depending on \( D[\mathcal{A}] \) we shall show that
the cohomology gives significant information as to the relations among the properties defined by $F(\phi)$. In particular this cohomology can be seen as the cohomology of measurements on $\mathcal{P}$. To illustrate this we start with cocycles. As usual $U[k] = U_k \cap U_m \cap \ldots \cup U_m$ and $\phi_k$ is defined on $U[k]$. A cocycle becomes an equation of the type $\sum_{k \text{ even}} F(\phi_k)(\xi) = \sum_{k \text{ odd}} F(\phi_k)(\xi)$ both sides restricted to $U = \cap_k U[k]$.

The simplest non trivial case being: $F(\phi_0) + F(\phi_2) = F(\phi_1)$. This indicates that $[\Lambda]$ has sufficient measures so the equation holds and that for any test schedule $\xi$ type $\Sigma$ contains $P$ the cohomology of $U \cap \xi \ni \phi_k \in U \cap \xi$ implies $U \cup \xi \ni \phi_k$ but $U \cap \xi \ni \phi_k$ contained in $U \cap \xi \ni \phi_k$ to a schedule in $U \cap \xi \ni \phi_k$. This then characterizes the objects of $Comb(\Lambda)$ in a functorial way.

This is just the start of a series of questions:

1. Can classes of sheaves $\Gamma(\Lambda)$ of $Comb(\Lambda)$ be classified by sufficient large sets of measures $\Lambda$ such that within $\Gamma(\Lambda)$ the $\mathcal{P}$ are defined by cohomology of their corresponding $\mathcal{P}_\emptyset$? As the cohomology groups $H^n(\mathcal{P}_\emptyset)$ are the classes of cocycles $\sum_{k \text{ even}} F(\phi_k)(\xi) = \sum_{k \text{ odd}} F(\phi_k)(\xi)$ that hold locally the equations become equations among the forms of $\Lambda$ that apply to $\mathcal{P}_\emptyset$ so characterizing the sheaves $\Gamma(\Lambda)$ in terms of "hyperplanes" on the multidimensional lattice $Z^\Lambda$.

2. Let $\Lambda_0 \subset \Lambda_0 \subset \ldots \subset \Lambda_n \subset ...$ be a sequence of forms such that $d^+_p : D[\Lambda_p] \to D[\Lambda_{p+1}]$ form exact sequences sheaves: $d^-$ chains, $d^+$ cochains. Can such structures be the basis of "resolutions" of a given $\mathcal{P}$ and so that the usual algebraic or algebraic geometry constructions [6], [11] can be applied to relational derived series $\mathcal{P}(\mathfrak{f})$ where $\mathcal{P} \leftarrow \mathfrak{f}(p)^\pm \to D[\Lambda_p]$?

4.3 Functorial properties

Given a logistic morphism $f : A_1 \to A_2$ and $\mathcal{P}_2$ a sheaf on $A_2$ defined for covering $U_2$, then the pullback of $\mathcal{P}_2$, $f^{-1}(\mathcal{P}_2)$ defined on $f^{-1}(U)$ is a sheaf on $A_1$. But it need not be in $Comb(\Lambda)$. For example, if $U \subset A_1$ is mapped by $f$ to a single element $b \in A_2$. If $\{b\}$ is in $U$ it is not clear how to interpret a schedule in $\mathcal{P}_2(b)$ to a schedule in $\mathcal{P}_1(U)$. For this reason this paper concentrates on cohomology as a covariant functor. In this case $f : A_1 \to A_2$ must preserve intersections: for $U_1, U_2, \ldots, U_n$ all in $U, f(U_1 \cap U_2 \cap \ldots \cap U_n) = f(U_1) \cap f(U_2) \cap \ldots \cap f(U_n)$. This ensures that the elements of cochains $C^n(\mathcal{P})$ map appropriately and $Sch(f)$ maps sheaves to sheaves giving the covariant functor $\hat{H}(f) : \hat{H}^n(\mathcal{P}_1) \to \hat{H}^n(\mathcal{P}_2)$. Preserving intersections is the same constraints on morphisms as being single-layer. (Proof: If there is a $U$ such that $f^{-1}(f(U)) \neq U$ then we can assume that there is a $V$ such that $f^{-1}(f(U \cup V)) = U$ so that $f(V) \subseteq f(U)$. If $U \cap V \neq \emptyset$ then as $V$ is not contained in $U$ then $V = V' \cup (U \cap V)$ where $V' \cap U = \emptyset$. Hence $f(U \cap V') = \emptyset$ but $f(U) \cap f(V') \neq f(V') \neq \emptyset$ so $f$ does not preserve intersections. But ($f$ is not single-layered implies $f$ does not preserve intersections) $\Leftrightarrow$ ($f$ preserves intersections $\Rightarrow f$ is single layered). Also single-layer implies preserves intersections: $f(U_1 \cap U_2) \subseteq f(U_1) \cap f(U_2)$ always. Assume single-layered. Let $x \in f(U_1) \cap f(U_2)$. Suppose $f(y) = x$ then $y \in f^{-1}(f(U_1) \cap f(U_2)) = f^{-1}(f(U_1)) \cap f^{-1}(f(U_2)) = U_1 \cap U_2$ as $f^{-1}$ preserves $\cap$. Therefore $f(y) = x \in f(U_1 \cap U_2)$ and so $f(U_1) \cap f(U_2) \subseteq f(U_1 \cap U_2)$. Thus single-layered implies preserves intersections and the two conditions are equivalent. This can be proved in any topos.)
Under the constraint that \( f \) preserve intersections the usual cohomology properties apply to the \( \check{\text{C}} \)ech cohomology \[30\]. In particular, for an exact sequence of sheaves the "long exact sequence" holds:

\[
\cdots \to \check{H}^n(P') \to \check{H}^n(P) \to \check{H}^n(P'') \to \check{H}^{n+1}(P') \to \cdots
\]  

(5)

In the following a combinatorial scheme \((A, P)\) is given with a covering \(U, U \in \mathcal{U}\) and \(B\) is a subobject of \(A\). Define presheaves \(P^B\) and \(P_B\) by

\[
P^B(U) = P(U) \quad U \cap B = \emptyset \quad (6)
\]

\[
P_B(U) = 0 \quad U \cap B \neq \emptyset \quad (7)
\]

and

\[
P_B(U) = P(U) \quad U \cap B \neq \emptyset \quad (8)
\]

\[
P_B(U) = 0 \quad U \cap B = \emptyset \quad (9)
\]

If we call the closure of \(B\) the set \( \bigcup_{b \in B} (S(b) \cup \{b\} \cup S^+(b)) \) then \(P^B\) zeroes everything inside this closure while \(P_B\) zeroes everything outside the closure.

These definitions allow the definition of a short exact sequence of presheaves defined on \(A\):

\[
0 \to P^B \to P \to P_B \to 0.
\]

Making the definitions

\[
\check{H}^n(B, P) = \check{H}^n(A, P_B)
\]

(10)

\[
\check{H}^n(A, B, P) = \check{H}^n(A, P^B).
\]

(11)

allows us to apply the long exact sequence to get relative cohomology:

\[
\cdots \to \check{H}^{n-1}(B) \to \check{H}^n(A, B) \to \check{H}^n(A) \to \check{H}^{n+1}(A, B) \to \cdots
\]

(Here the sheaves are implicit). These results follow from standard (co)homological algebra and will not be proved here.

The Excision theorem also has a version here. In topology, given \(B_1 \subset B \subset A\) the Excision theorem requires that the closure of \(B_1\) is in the interior of \(B\). For combinatorial system with \(B_1 \subset B \subset A\) we require that all \(b \in B_1\) \(S(b) \cup S^+(b) \subseteq B\) then

\[
\check{H}^n(A \setminus B_1, B \setminus B_1) = \check{H}^n(A, B).
\]

(12)

These results show that the cohomology developed here justifies it claim as a such a functor even though it is covariant on its domain category.

**Part II**

**Applications: manufacturing and supply networks**

This part of the paper addresses applications in more detail. Because of length of the paper only one class of applications is selected to illustrate the following points.

25
1. The application of combinatorial systems that uses the full set of definitions including replenishment.

2. The application of the $[L, K]$ in real circumstances.

3. The economic importance of the concepts.

For these reasons the section is on manufacturing networks. Advanced technology manufacturing is characterized by the scale of detailed documentation\textsuperscript{5}, its highly systematized processes (algorithms) to ensure specifications are met and elaborate connections of engineering, physical, chemical and marketing reasons for doing things.

## 5 Manufacturing and Supply Networks.

Manufacturing networks, also called supply networks and (most commonly and misleadingly called) supply chains were the first motivating examples for combinatorial systems\textsuperscript{7}. These networks marshal the material (types) and expertise (actions) required to make something. They are the international face of the integration of expertise across the world; they can be defined very widely indeed and are of paramount importance to modern economies. The scale of manufacturing networks is considerable: according to Ford CEO Alan Mulally (quoted in 'Time' 6th Sept. 2010 p.28) a car has 10,000 parts and an airliner 4 million.

Manufacturing networks have an extensive history of mathematical modeling. There are many themes or perspectives. The majority of this research has been in optimization and simulation and many recent surveys, \cite{31, 2, 26, 9, 22, 17, 27}, give a guide to that literature. A related perspective considers the design of supply chains as a social structure of obligations and is motivated by Game Theory and procurements strategies \cite{3}. Optimal solutions, simulation and contractual games will give structures to aim for. Attaining and keeping the optimal conditions can be extremely difficult especially when managing very large numbers of variables such as supplier prices and raw material transport costs. The difficulty of controlling variables and the growth of inter-company IT networks has resulted in studies of the effects information sharing and show that information sharing reduces the fluctuations that arise in response to changing demand \cite{15, 4}. This has prompted the approach characterized as the collaborative planning, forecasting and replenishment or "CPFR" \cite{18}.

We take a manufacturing network to be a combinatorial module interpreted as follows.

1. $A$ is the set of organizations, called suppliers, involved in the manufacture or movement of goods to a final destination $a^*$, the maximum element of $A$.

2. $T_{pc}(a)$ is an action $\alpha(a) : W(a) \times Typ(a) \rightarrow Typ(a)$ where

   (a) The objects of $Typ(a)$ in a supply network are inventory states $q = \sum_i x_i e_i$, $x_i \in \mathbb{Z}$ where the $e_i$ are the objects of $T_{pc}(a)$. In this way the states of $a$ correspond to the sets of items that are necessary for manufacturing successive assemblies of components for a final product.

\textsuperscript{6}"When the documentation is as heavy as the aircraft you know you are getting it right" Quoted from TV series "How to make a" episode of 'How to make the wing of a super-jumbo (A380)' SBS (Australia) 18/08/2012 7.30p.m.

\textsuperscript{7}Manufacturing networks are most relevant here because of the emphasis on algorithms. Supply networks typically emphasis inventory and logistics.
(b) \( W(a) \) is the ring generated by \( act(a) \) of processes required to accomplish the manufacture or transport of goods. \( act(a) \) is therefore the monoid of actions that produce "work in progress" inventory items.

3. \( \beta(a, b) \) with \( b \) in \( S(a) \) corresponds to the "bill of materials" function from \( a \) to a supplier of \( a \).

4. \( L \) is the lead-time between the receipt of an order and the dispatch of goods when the supplier has to manufacture the item from starting inventory items.

As most actions \( w \in W(a) \) deplete \( Typ(a) \) replenishment is required. Keeping track of this is the reason for \( Typ(a) \) being a \( \mathbb{Z} \) module. The replenishment logic of a supplier \( a \) vary from the simple: if \( e \) goes below an amount \( q_{\text{min}}(e) \) order enough to get back to \( q_{\text{max}}(e) \), to elaborate and strategically important algorithms for forecasting, automated purchasing and sharing information with relevant suppliers in the network. The significance and variety of replenishment methods cannot be underestimated. It cannot be given due treatment here.

Let \( \bar{T} \) be an interval of \( \mathbb{N} \). In manufacturing networks it is convenient to represent a schedule as a map \( \xi : \bar{T} \rightarrow W(a) \) where \( \gamma(a) \) is the maximum number of \( w \in W(a) \) that can occur at any one time slot. This corresponds to the previous definition by \( s^{-1}(t) = (\xi)(t) \).

Reversing time in a supply network produces a "distribution plan". In this case a sequence of distributions start at \( a^* \) and then get either unpacked and distributed further or used at each \( a \in A \). Examples include the distribution of oil from a super-tanker and the distributions of food and drugs for humanitarian aid.

**Definition.** A scenario is a schedule that fulfills the sequence of requests or orders to a manufacturing network. It starts with

1. a sequence of orders or "requests" \( r_1, r_2, ..., r_n \) issued at time \( t_1, t_2, ..., t_n \),

2. a starting state \( q(t_0) \in Typ(A) \) with \( t_0 \leq t_1 \),

3. the requests are to be fulfilled at \( t'_1, t'_2, ..., t'_n \) respectively. Alternatively this can be stated that there is a requirement \( \sigma_r : T \rightarrow Typ(a^*) \) where \( T \) is time,

The solution of the scenario is the schedule that can be constructed within the logic of the manufacturing network to deliver \( r_i \) on or before the \( t'_i \).

A scenario is a theorem that a schedule can connect a starting state to a final result. In manufacturing networks this can cover many depletion cycles in which all inventory is used up many times and has to be replenished. This means that schedules for networks that require sustained operations are much more complex than combinatorial systems that do not have depletion of objects.

\( scn(A) \) is the set of scenario requests or requirements that can be formulated in \( A \).

If \( \sigma \) is a scenario and \( \xi \) is a schedule that solves it we write \( \xi \Rightarrow \sigma \). Not all scenarios can be so solved, some are beyond the networks capability. It is the boundary between success and failure of a manufacturing network that best defines its capacity. The next section applies the results of section 3 on \( P_\Sigma \) and \( K_\Sigma \) where \( K \) becomes a new sheaf of scenarios. However the relationship "\( \Rightarrow \)" which corresponds to \( \Sigma \) is not defined directly in terms of sheaves so this has to be done.
5.1 Capability: Scenarios and Proofs

$\mathcal{U}$ is the standard cover of $\mathcal{A}$ and $\mathcal{P} \in \text{Comb}(\mathcal{A})$. $\text{Typ}(\mathcal{U}) = \oplus \text{Typ}(a)$ with each $\text{Typ}(a)$ a $\mathcal{Z}$ module defines a $\mathcal{Z}$ module valued pre-sheaf that is trivially a sheaf. Given a scenario $\sigma$ let $\sigma(a)$ be the bill of materials required at $a$ that is necessary for $\sigma$ to be satisfied.

**Definition** The sheaf of scenarios, denoted $\text{Scn}(\mathcal{A})$ is defined in a number of steps as follows.

1. Define the "inventory sheaf" $\text{Inv}$ on $\mathcal{A}$ as $\text{Inv}(a) = \text{Typ}(a) \times \mathcal{Z}$ which is a sheaf in $\mathcal{Z}$ modules. Here $\mathcal{Z}$ is the trivial constant sheaf $\mathcal{Z}(\mathcal{U}) = \mathcal{Z}$. Because of the independence of states among the suppliers $\text{Inv}$ is trivially a sheaf.

2. Give a scenario $\sigma$ define $(\sigma(a), t)$ to be the types required in $\text{Typ}(a)$ at time $t$. The condition on this is that the types $\sigma(a')$ must be in $\text{Typ}(a')$ at times $t_i$ as in the request $\sigma_r$ of the scenario and if $(\sigma(a), t), b \in S(a)$ and $(\sigma(b), t')$ then $t' < t - 1$. This gives a section:

$$\text{scn}(\mathcal{A}) \xrightarrow{\sigma} \text{Inv}; \xi(U) : \sigma \mapsto \{ (\sigma(a), t) \in \text{Inv}(a) \mid a \in \mathcal{U}, t \text{ constrained as above} \}$$

3. $\text{Scn}(\mathcal{A})(\mathcal{U})$ is the image of $\xi(U)$ and is clearly a presheaf.

$\text{Scn}(\mathcal{A})$ is in fact a sheaf because of weak constraints on sections. **Definition.** The addition of scenarios. For any $\mathcal{U} \in \mathcal{A}$ and $\sigma_1$ and $\sigma_2$ in $\text{Scn}(\mathcal{U})$ define $\sigma_1 + \sigma_2$ as the sum of the functions $\sigma_i : T \rightarrow \text{Inv}(\mathcal{U})$.

**Proposition 5.1.** The addition of scenarios makes $\text{Scn}$ a group valued sheaf.

The proof is left to the reader.

$\text{Scn}(\mathcal{A})$ has very few constraints on its values other than times and these are minimal constraints. It gives us very little information about $\mathcal{A}$. The times associated with scenario requirements can vary from impossible for any known technology to pointlessly long.

Define $\text{inv} : \mathcal{P} \rightarrow \text{Inv}$ as follows: given $\xi \in \mathcal{P}(\mathcal{U})$, for all $a \in \mathcal{U}$, $\text{inv}(\xi)(a)$ is given by the set $(q(a), t)$ if $\xi$ leaves $\text{Typ}(a)$ in the state $q(a)$ at time (slot) $t$. If $b \in S(a)$ and $\{a, b\} \subset \mathcal{U}$ the state that $\xi$ leaves $\text{Typ}(b)$ in, $\text{inv}(\xi)(b)$, has to precede that of the state of $\text{Typ}(a)$. This is a condition on $\xi$ defined for $\mathcal{U}$. The map $\text{inv}$ is a sheaf map as is easily seen; the conditions on patching schedules defined on a set of $\mathcal{U}$ is more stringent than that of patching states in $\text{Inv}$.

$\text{Inv}$ has a sheaf relation $[\leq]$ where $[\leq](q, t; q', t') := (q \leq q') \land (t = t')$. (This can be built from $\leq \rightarrow \text{Typ} \times \text{Typ}$ and the diagonal map $\mathcal{Z} \xrightarrow{(1, 1)} \mathcal{Z} \times \mathcal{Z}$)

Define a sheaf relation $\Rightarrow \rightarrow \mathcal{P} \times \text{Scn}(\mathcal{A})$ as the pullback

$$\Rightarrow \xrightarrow{\text{inc}} \mathcal{P} \xrightarrow{\text{proj}_2 \circ (\text{inv}, \text{inv})} \text{Scn} \xrightarrow{(\text{proj}_2 \circ (\text{inv}, \text{inv}))} [\leq].$$

This gives

$$\Rightarrow (\mathcal{U}) =_{\text{defn}} \{ (\xi, \sigma) \mid (\xi \in \mathcal{P}(\mathcal{U})) \land [\leq] (\xi(U)(\sigma), \text{inv}(\xi)) \}.$$
That is

$$\Rightarrow (U) =_{defn} \{(\xi, \sigma) \mid (\xi \in \mathcal{P}(U)) \land (s(U)(\sigma) \leq inv(\xi))\}.$$ 

This means that if $a^* \in U$ $inv(\xi)$ must equal or exceed the quantities in the requests of $\sigma$ at the time that they are required and the times in $s(U)(\sigma)$ match those in $inv(\xi)$ and so correspond to a real schedule in $U$.

As $[\Rightarrow, Scn]$ is in the form of $[\mathcal{L}, \mathcal{K}]$: writing $\Rightarrow (\xi, \sigma)$ as $\xi \Rightarrow \sigma$ we can define:

1. The set-valued sheaf $Scn_{\Rightarrow}$ given by
   
   (a) For $\xi \in \mathcal{P}(U)$, $S(\xi) =_{defn} \{\sigma \in Scn(U) \mid \xi \Rightarrow \sigma\}$
   
   (b) $Scn_{\Rightarrow}(U) =_{defn} \{S(\xi) \mid \xi \in \mathcal{P}(U)\}$

2. The set-valued sheaf $\mathcal{P}_{\Rightarrow}$ given by
   
   (a) For a scenario $\sigma$ and $U \subseteq A$, $Z(\sigma) =_{defn} \{\xi \in \mathcal{P}(U) \mid \xi \Rightarrow \sigma(U)\}$.
   
   (b) For $U \in \mathcal{U}$, $\mathcal{P}_{\Rightarrow}(U) =_{defn} \{Z(\sigma)(U) \mid \sigma$ is a scenario $\}$.

Note that if $\xi$ and $\xi'$ imply the same set of local scenarios then $S(\xi) = S(\xi')$.

Assuming that the times $\{t_1, t_2, ..., t_n, ...\}$ and $\{t'_1, t'_2, ..., t'_n, ...\}$ come from a discrete bounded and therefore finite set $\text{scn}(A)$ is a finite set and ordered by inclusion: $\sigma_1 < \sigma_2$ if the requests of $\sigma_1$ are a subset of the requests of $\sigma_2$ but the times of delivery are the same. This means that if $\xi_2$ can satisfy $\sigma_2$ there is a $\xi_1$ that satisfies $\sigma_1$ and $\xi_1 \subseteq \xi_2$. With scenarios and schedules finite partially ordered sets they are order complete and we can apply Theorem 3.3.

**Corollary 5.1.** If the partial orders given by inclusion on the sets $Z(\sigma)$ and $S(\xi)$ are complete then there are sheaf maps $\downarrow: \mathcal{P}_{\Rightarrow} \to Scn_{\Rightarrow}$ and $\uparrow: Scn_{\Rightarrow} \to \mathcal{P}_{\Rightarrow}$ that are inverses.

Note that if $\sigma$ cannot be satisfied by any schedule on any covering set of $\mathcal{P}$ then $Z(\sigma)$ is empty. Furthermore there is no $\xi$ such that $S(\xi)$ contains $\sigma$.

**Definition.** Group addition for the sheaves $\mathcal{P}_{\Rightarrow}$ and $Scn_{\Rightarrow}$: for $\mathcal{P}_{\Rightarrow}$ define an addition, denoted by $+$, at each $U$ by the local relations $Z(\sigma_1) + Z(\sigma_2) =_{defn} Z(\sigma_1 + \sigma_2)$. Similarly for $Scn_{\Rightarrow}$ define addition denoted by $+$ and defined at each $U$ by the local relations $S(\xi_1) + S(\xi_2) =_{defn} S(\xi_1 + \xi_2)$.

As defined these are sheaves as they are defined in terms of schedules and scenarios which are already sheaf sections in existing sheaves. As $S(\xi_1 + \xi_2)$ contains both $S(\xi_1)$ and $S(\xi_2)$ while $Z(\sigma_1)$ and $Z(\sigma_2)$ both contain $Z(\sigma_1 + \sigma_2)$ the respective definitions of addition do not indicate set identification. This is added structure.

**Definition.** A manufacturing supply network $\mathcal{N}$ operates on make to schedule quantity if orders are accepted only in quantities that fit a range of schedules. This includes "assemble to order". This means that whenever a set of orders (scenarios) $\sigma_i$ with $\sigma_i(a) = \sum n_{ij}c_{ij}$ have make to order schedules $\xi_i$ then there are no common components or processes that can be batched or combined. In this case $\Sigma \sigma_i(a)$ has a least schedule $\xi = \Sigma \xi_i$ in $\sum(Z(\sigma_i))$ giving $\Sigma(\sum(Z(\sigma_i))) = \sum(\Sigma(Z(\sigma_i)))$.

The effect of making to schedule quantity with schedules having minimum orders is illustrated by the following example. Suppose all schedules $\xi$ produce a certain

---

8"Lean manufacturing" has this linearity as a goal. It is the same as "scaling without drama". A similar concept is minimum order quantities to fit jumps in scheduling.
item e in multiples of 5. \( \sigma_1 \) wants 6 of e and \( \sigma_2 \) wants 11. \( \xi_1 + \xi_2 \) produce 25 between them but \( \xi \) need only produce 11. If \( A \) operates on minimum order sizes then \( \xi_1 + \xi_2 \) produces the same amount as \( \sigma_1 + \sigma_2 \). This is equivalent to \( \xi_2 \), producing exactly what \( \sigma \) wants and the same for \( \xi \).

**Proposition 5.2.** If \( A \) operates on make to schedule quantity then \( j \) and \( f \) are homomorphisms

**Proof.** Write \( j(Z(\sigma)) \) as \( S(\xi) \) where \( \xi \) is the least \( \xi \) with \( \xi \Rightarrow \sigma \). Then \( j(Z(\sigma_1) \vdash Z(\sigma_2)) = j(Z(\sigma_1 + \sigma_2)) \) which is to equal \( j(Z(\sigma_1)) \vdash j(Z(\sigma_2)) \).

\[
j(Z(\sigma_1)) \vdash j(Z(\sigma_2)) = S(\xi_1) \vdash S(\xi_2) = S(\xi_1 + \xi_2) \quad \text{(by definition)}.
\]

\( j(Z(\sigma_1 + \sigma_2)) = S(\xi) \) where \( \xi \) is the greatest lower bound with \( \xi \Rightarrow \sigma_1 + \sigma_2 \). As \( \xi_1 \Rightarrow \sigma_1 \), it is natural to identify \( \xi_1 + \xi_2 \) with \( \xi \) in which case \( j \) would have addition. \( A \) makes to schedule quantities implies item for item \( \xi \) matches \( \xi_1 + \xi_2 \) and so \( S(\xi) = S(\xi_1 + \xi_2) \) and \( j \) preserves the addition. That \( j \) takes 0 to 0 and inverses to inverses is obvious so \( j \) is a homomorphism.

Again write \( f(S(\xi)) = Z(\hat{\sigma}) \) where \( \hat{\sigma} \) is the least upper bound of \( \sigma \) with \( \xi \Rightarrow \sigma \).

We want to show

\[
f(S(\xi_1) \vdash S(\xi_2)) = f(S(\xi_1 + \xi_2)) \quad \text{is equal to} \quad f(S(\xi_1)) \vdash f(S(\xi_2)).
\]

But

\[
f(S(\xi_1)) \vdash f(S(\xi_2)) = Z(\hat{\sigma}_1) \vdash Z(\hat{\sigma}_2).
\]

\( \hat{\sigma}_1 \) is the least upper bound for \( \xi_1 \). Hence \( \xi_1 + \xi_2 \Rightarrow \hat{\sigma}_1 + \hat{\sigma}_2 \). Using the make to schedule quantities requested in scenarios and schedules the least upper bound in both cases is the same and so \( f(S(\xi_1) \vdash S(\xi_2)) = Z(\hat{\sigma}_1) \vdash Z(\hat{\sigma}_1) \). The remaining details to show \( f \) is a homomorphism are clear.

The condition "make to schedule quantities" is crucial here as the proof indicates.

**Proposition 5.3.** If \( A \) operates on make to schedule quantities the cohomology of the sheaves \( \mathcal{P}, \mathcal{P}_\Rightarrow \) and \( \text{Scn}_\Rightarrow \) are identical.

**Proof.** In \( \text{C}^n(\mathcal{P}_\Rightarrow) \) an \( n \)-cocycle requires

\[
Z(\sigma_0) \vdash Z(\sigma_2) \vdash \ldots \vdash Z(\sigma_m) = Z(\sigma_1) \vdash Z(\sigma_3) \vdash \ldots \vdash Z(\sigma_{m'})
\]

where \( m, m' \) are \( n \) and \( n - 1 \) depending on whether \( n \) is even or odd. By definition this is \( Z(\sigma_0 + \sigma_2 + \ldots + \sigma_m) = Z(\sigma_1 + \sigma_3 + \ldots + \sigma_{m'}) \) and corresponds to \( \sigma_0 + \sigma_2 + \ldots + \sigma_m \) in \( \text{C}^n(S) \). It also corresponds to

\[
S(\xi_0) \vdash S(\xi_2) \vdash \ldots \vdash S(\xi_m) = S(\xi_1) \vdash S(\xi_3) \vdash \ldots \vdash S(\xi_{m'})
\]

that is \( S(\xi_0 + \xi_2 + \ldots + \xi_m) = S(\xi_1 + \xi_3 + \ldots + \xi_{m'}) \).

As the schedules effectively match the scenarios we have

\[
\xi_0 + \xi_2 + \ldots + \xi_m = \xi_1 + \xi_3 + \ldots + \xi_{m'}.
\]
which gives a cocycle in $C^n(\mathcal{P})$.

A similar argument applies to coboundaries. The "make to schedule quantities' condition is really a statement asserting a close match between scenarios and schedules; it means that any schedule can appear as a $S(\xi)$ as we simply adjust scenarios to require it. This way the three cohomologies are the same.

The importance of this result is when it fails. The failure of make to schedule quantities or any of its equivalent matching statements is asserted whenever this proposition fails. It is a way of measuring the difference between the operations of $A$ and a supply chain that adjusts schedules to orders.

How bad can the difference difference between $\mathcal{P}_{\omega}$ and $Scn_{\omega}$ be? Many factory operations can work on a 8 hour shift, an 8 hour shift with overtime, two shifts, two shifts with overtime and three shifts. Scenarios are then batches of orders added so that they are a collective order load $\sigma = \sum_{i=1}^{n} \sigma_i$ and the scheduling is matched as closely as possible. This is similar to public transport systems that have peak hour and off-peak services which have significant gaps between the total order and coordination then as things fail we can expect the cohomology to become as "Fragmentation Theory". If a soft sheaf has all but the zero-th cohomology trivial. It indicates that all sections can be interpreted as $Inv(A)$. When a manufacturing network, $A$, is faced with a stream of orders that cannot be delivered on time certain parts of $A$ will fall below the required capacity while other parts will have sufficient capacity. The study of this phenomena is referred to here as "Fragmentation Theory". If a soft sheaf indicates the greatest level of internal order and coordination then as things fail we can expect the cohomology to become more complicated. In this way there is an analogy with phases in physical media that are characterized by correlation [1] and fragmentation is a theory of the phases of lost correlation as measured by cohomology.

Let $\tau$ be an order to $a^\tau$ to deliver $r_i$ goods no later than $t_i$ and let $q \in Inv(A)$ be the starting inventory given by $q(a)$ for each $a \in A$. That is, a scenario $\sigma(\tau,q)$.

**Definition.** Define $Sch(\tau,q)$ to be the group-valued sheaf generated by schedules that satisfy $\sigma(\tau,q)$.

As a sheaf $Sch(\tau,q)(U)$ is the set of schedules generated by $Z(\sigma(\tau,q)) \in Sch_{\omega}(U)$ and so we can write $Sch(\sigma(\tau,q)) \Rightarrow \sigma(\tau,q)$. Although $Z(\sigma(\tau,q))$ is a semi-group, the sum of any two schedules that satisfy $\tau,q$ again satisfy the scenario, $Sch(\tau,q)(U)$ is the smallest group valued sheaf containing $Z(\sigma(\tau,q))$. Because the subsheaves $Sch(\tau,q)$ of $Sch$ are concerned with time constrained schedules they cannot be expected to be soft. Also if $\xi$ is a solution to $\sigma(\tau,q)$ then any $\xi' \geq \xi$ is a solution to $\sigma(\tau,q)$.

5.2 Fragmentation Theory

When a manufacturing network, $A$, is faced with a stream of orders that cannot be delivered on time certain parts of $A$ will fall below the required capacity while other parts will have sufficient capacity. The study of this phenomena is referred to here as "Fragmentation Theory". If a soft sheaf indicates the greatest level of internal order and coordination then as things fail we can expect the cohomology to become more complicated. In this way there is an analogy with phases in physical media that are characterized by correlation [1] and fragmentation is a theory of the phases of lost correlation as measured by cohomology.

Let $\tau$ be an order to $a^\tau$ to deliver $r_i$ goods no later than $t_i$ and let $q \in Inv(A)$ be the starting inventory given by $q(a)$ for each $a \in A$. That is, a scenario $\sigma(\tau,q)$.

**Definition.** Define $Sch(\tau,q)$ to be the group-valued sheaf generated by schedules that satisfy $\sigma(\tau,q)$.

As a sheaf $Sch(\tau,q)(U)$ is the set of schedules generated by $Z(\sigma(\tau,q)) \in Sch_{\omega}(U)$ and so we can write $Sch(\sigma(\tau,q)) \Rightarrow \sigma(\tau,q)$. Although $Z(\sigma(\tau,q))$ is a semi-group, the sum of any two schedules that satisfy $\tau,q$ again satisfy the scenario, $Sch(\tau,q)(U)$ is the smallest group valued sheaf containing $Z(\sigma(\tau,q))$. Because the subsheaves $Sch(\tau,q)$ of $Sch$ are concerned with time constrained schedules they cannot be expected to be soft. Also if $\xi$ is a solution to $\sigma(\tau,q)$ then any $\xi' \geq \xi$ is a solution to $\sigma(\tau,q)$.

---

5A soft sheaf has all but the zero-th cohomology trivial. It indicates that all sections can be extended to global sections. Also known as a "flasque" sheaf.
10interpreted as $Inv(a^\tau)$ will have $r_i$ goods available at $t_i$. 31
**Definition** The scenario sub-category of $\text{Comb}(A)$. The objects of this category are the sheaves $\text{Sch}(r,q)$ and morphisms are embeddings $\text{Sch}(r_1,q_1) \hookrightarrow \text{Sch}(r_2,q_2)$.

We shall say that $A$ is "normal" if the following is satisfied:

$$r_1 \geq r_2 \text{ and } q_1 \leq q_2 \Rightarrow \text{Sch}(r_1,q_1) \hookrightarrow \text{Sch}(r_2,q_2).$$

(13)

This need not be the case. There can be products that are easier to make in large amounts by utilizing batch processes that can use economies of scale that are not available for smaller amounts. Small amounts cannot use the batch processes and so take more time to "craft by hand" than manufacture using robotized factories.

**Theorem 5.2. (The coboundary influence theorem)** If $\mathcal{P}_i, i = 1, 2$ are two subsheaves of $\text{Sch}$ and $j: \mathcal{P}_1 \hookrightarrow \mathcal{P}_2$ then the induced homomorphism $\hat{H}(j): \hat{H}(\mathcal{P}_1) \rightarrow \hat{H}(\mathcal{P}_2)$ is defined and

$$\ker(\hat{H}(j)) = (Z^k(\mathcal{P}_1) \cap (C^k(j))^{-1}(B^k(\mathcal{P}_2)))/B^k(\mathcal{P}_1)$$

(14)

where $C^k(j)$ is the induced map $C^k(\mathcal{P}_1) \rightarrow C^k(\mathcal{P}_2)$.

**Proof:** For any set of schedules defined on $\mathcal{P}_1$ they are also in $\mathcal{P}_2$.

This means

$$\begin{array}{ccc}
C^{-1}(\mathcal{P}_1) & \xrightarrow{\partial^{-1}} & C^0(\mathcal{P}_1) \\
\downarrow & & \downarrow \\
C^{-1}(\mathcal{P}_2) & \xrightarrow{\partial^{-1}} & C^0(\mathcal{P}_2)
\end{array}$$

commutes. A cocycle in $C^k(\mathcal{P}_1)$ will satisfy the same equation in $C^k(\mathcal{P}_2)$ and so $Z^k(\mathcal{P}_1)$ is a subgroup of $Z^k(\mathcal{P}_2)$ and similarly $B^k(\mathcal{P}_1)$ is a subgroup of $B^k(\mathcal{P}_2)$.

If $z \in \hat{H}^k(\mathcal{P}_1)$ and $z \neq 0$ then $z$ represents an element in $Z^k(\mathcal{P}_1)$ that is not in $B^k(\mathcal{P}_1)$. Hence

$$\hat{H}^k(j)(z) = 0 \Leftrightarrow z \in B^k(\mathcal{P}_2)$$

so that

$$\ker(\hat{H}^k(j)) = \hat{H}(j)^{-1}(B^k(\mathcal{P}_2))/B^k(\mathcal{P}_1)$$

and $(\hat{H}(j)^{-1}(B^k(\mathcal{P}_2)) = Z^k(\mathcal{P}_1) \cap (C^k(j))^{-1}(B^k(\mathcal{P}_2))$ which gives the result.

**Definition.** The frontier of $A$ is the set $(r,q)$ such that for any pair $(\delta r, \delta q) \in \text{Typ}(a^*) \times \text{Typ}(A)$, $\delta r$ and $\delta q$ positive, $\hat{H}^0(\text{Sch}(r,q))$ is non trivial and $\hat{H}^0(\text{Sch}(r + \delta r, q - \delta q)) = 0$.

The frontier divides the scenario space of elements $(r,q)$ into demand that can be accomplished and demand for which there is no global schedule.

To simplify the discussion set $q = 0$ so "initializing" with an empty inventory and hence the scenario is in terms of the longest initial lead-times for $A$. Let $\mathcal{P}(\xi) = \text{Sch}(\xi,0)$ and $\hat{H}^0(\mathcal{P}(\xi)) \neq 0$. If $\Psi(r) = \{\delta r \mid \hat{H}^0(\mathcal{P}(\xi + \delta r)) = 0\}$ where $\delta r \in \text{Typ}(a^*)$ then the smallest $\delta r \in \Psi(r)$ to change the cohomology from soft to have $\hat{H}^0(\mathcal{P}(\xi + \delta r)) = 0$ is the fragmentation gradient.

At the frontier an order is as big as can be with a given, achievable lead-time. The fragmentation gradient gives the increase in size of the order for which things become unpredictable.
The difference between $dim(\tilde{H}^n)$ (as a $\mathbb{Z}$ module) for each $n$ and $dim(Z^n)$ for each $n$ provides a rough measure of the coordination of schedules.

**Definition.** The **cocycle mark** of a sheaf $\mathcal{P}$ denoted $mk_Z$ is the formal sum

$$mk_Z(\mathcal{P}) = \sum_{n \geq 0} dim(Z^n(\mathcal{P})).t^n$$  

and the **class mark** of $\mathcal{P}$ denoted $mk_H$ is the formal sum

$$mk_H(\mathcal{P}) = \sum_{n \geq 0} dim(\tilde{H}^n(\mathcal{P})).t^n$$

Example: With rather weak constraints on the duration of schedules $Sch$ can be shown to be soft. Any production of any goods can be extended to be an order of something; $mk_Z(Sch) - mk_H(Sch) = mk_Z(Sch) - dim(\tilde{H}^0(Sch))$.

If $Sch(r_1) \hookrightarrow Sch(r_2)$ (as when $r_1 > r_2$) then the coboundary influence theorem gives a description of the way $Sch(r_2)$ can grow. If for a given $k$, $B^k(Sch(r_2))$ is large then a non-zero kernel is possible giving a measure of the change in cohomologies. For a sufficiently large $\delta r$ eventually $Sch(\tau + \delta r)$ cannot deliver $\tau + \delta r$ in the time required by the scenario $\sigma(\tau)$. The extreme case is:

**Definition.** A request $\tau + \delta r$ is on the **disordered boundary** when

$$mk_Z(Sch(\tau + \delta r)) = mk_H(Sch(\tau + \delta r)).$$

This means that whenever schedules coincide locally it happens because of local circumstances that are not derived from any larger structure. What order there is is local. Coming in from the disordered boundary the sheaves get bigger and bigger until there are enough coboundaries to kill off the cocycles. The coboundary influence theorem is applicable here.

**Proposition 5.4.** If $(\tau + \delta r)$ is on the disordered boundary and $\delta r > \delta r'$ and $j : Sch(\tau + \delta r) \hookrightarrow Sch(\tau + \delta r')$ then

$$ker\tilde{H}^k(j) = Z^k(Sch(\tau + \delta r)) \cap B^k(Sch(\tau + \delta r'))$$


**Proof:** This is really a corollary of the coboundary influence theorem as $B^k(Sch(\tau + \delta r)) = 0$. The kernel measures how much the cohomology diminishes and more order is created as coboundaries indicate the growth of order. The set of all $\delta r$ between the fragmentation boundary and the disordered boundary can be characterized by the cohomology of the $Sch(\tau + \delta r)$.

6 **The Coordination of Change**

In this section the sheaf $\mathcal{G}_\mathcal{L}$ (introduced in section 3) is applied to manufacturing networks. This object is a preoccupation of managers of jobbing shops across the world. Not, of course, in this form.
6.1 Groupoid Classes and the \( J(\mathcal{G}, \mathcal{P}) \) functor

Section 3.3 presented a groupoid action on a sheaf of schedules, \( s, t : \mathcal{G} \cong \mathcal{P} \) in terms of the \([\mathcal{E}, \mathcal{K}]\) construction. This produced a corresponding sheaf \( \mathcal{P}_{\mathcal{E}} \). In this section \( \mathcal{G}_{\mathcal{E}} \) will be our main interest. To emphasize this as a functor from groupoid fibrations over sheaves in \( \text{Comb}(\mathcal{A}) \) we shall write \( \mathcal{G}_{\mathcal{E}} \) as \( J(\mathcal{G}, \mathcal{P}) \) and \( J(\mathcal{G}) \) when the \( \mathcal{P} \) is understood. \( J(\mathcal{G}, \mathcal{P}) \) is contravariant in groupoids and covariant in the the objects of \( \text{Comb}(\mathcal{A}) \): \( \mathcal{G}' \xrightarrow{\mathcal{h}} \mathcal{G} \Rightarrow \mathcal{P} \) pulls back to groupoid fibration \( s \circ h, t \circ h : \mathcal{G}' \cong \mathcal{P} \). Covariance in \( \mathcal{P} \) is easily shown. From section 3.3 \( J(\mathcal{G}, \mathcal{P}) \) is given by

1. \( G(U)(\xi) = \{ \xi' \in \mathcal{P}(U) \mid \exists g \in \mathcal{G}(U)(g \neq 1) \wedge (sg = \xi) \wedge (tg = \xi') \} \) and so
2. \( J(\mathcal{G}, \mathcal{P})(U) \equiv \mathcal{G}_{\mathcal{E}}(U) = \{ G(U)(\xi) \mid \xi \in \mathcal{P}(U) \} \).

An arbitrary section of \( J(\mathcal{G}, \mathcal{P}) \) with be written \( G(\xi) \).

The importance of groupoid actions on schedules is that schedules are frequently adjusted, indeed a serious concern of factories throughout the world. Adjustments to schedules can be composed and in this context reversed. Adjustments to one schedule need not apply to another. This makes them a groupoid in the sense Brandt (see [13],[8],[12] for information on groupoids) and so generalize to groupoid fibrations as in section 3.3.

There are many classes of groupoids that are relevant to changing schedules in an industrial setting.

1. Adjustments for schedules that do not change elements of \( w \in W(a) \) that use large amounts of an expensive and not always available resource. Let \( u : W(a) \rightarrow \mathbb{N} \) be a ranking of factory resources required to accomplish \( w \) in a given amount of time. Define the groupoid \( G(u^{-1}(k)) \) as the set of changes that leave \( \{ w | u(w) \geq k \} \) unchanged within a schedule. This is clearly a groupoid and corresponds to the practice of changing schedule around the "large" operations such as batch jobs or processes that take up disproportionate amount of factory resources.

2. Adjustments associated with a groupoid \( G^+ \) which is defined by the property that \( \forall g \in G^+ \) there is a \( t' \in T \) such if \( t \leq t' < \max \{ t \mid t \in \text{dom}(\xi) \} \) then \( g\xi(t) = \xi(t) \). This allows a schedule to be changed while in process. In this case an adjustment \( \xi \xrightarrow{g} \zeta \) might divide into adjustments earlier and later than a given time, such as the time of arrival of newly scheduled goods. Then \( g\xi(t \leq t_a) = \xi_1 \) and \( g\xi(t > t_a) = \xi_2 \).

3. Schedules that depend on a set of variables or parameters that can be changed are a source of groupoids and their associated equivalences classes. The groupoid acts on schedules through the schedule’s dependence on parameters that can be changed. If \( \xi = \xi(v_1, v_2, ..., v_n) \) where \( v_i \in V_i \) such that if \( g \) acts on each \( V_i \) then \( g \) changes \( \xi \) by the action \( g\xi(v) = \xi(g(v)) \) for some \( v \in \oplus V_i \). For this to make sense there must be a \( \xi' = \xi(g(v)) \). Typical parameters include times to reach various production stages, costs of labor, costs of using fast or slow machines, slower or faster processing time that come from using different input materials which eventually give the same final output. In particular not all parameter changes have a schedule associated with them. Groupoids acting on such parameterize schedules can be decomposed into sub-groupoids that give the independence of subsets of parameters.
Definition of the group valued sheaf of adjustment classes

The structure of a given $G(\xi)$ can be analyzed in terms of its subgroupoids. If $g : \xi_1 \to \xi_2$ there is an associated set function $\tilde{g} : G(\xi_2) \to G(\xi_1)$ given by $\tilde{g}(h) = h \circ g$. It is not a groupoid homomorphism [12] but allows the decomposition of groupoids in a way that reflects the hierarchy of equivalence relations among the schedules connected by adjustments. For example given

$$\xi_1 \xrightarrow{g_1} \xi_0 \xrightarrow{g_2} \xi_2$$

where $g_1$ and $g_2$ are the only adjustments from $\xi_0$ to $\xi_1$ and $\xi_2$ then $G(\xi_0)$ decomposes as $G(\xi_1) \sqcup G(\xi_2)$. Furthermore any equivalence relation between a $\xi'$ connected to $\xi_1$ and $\xi''$ connected to $\xi_2$ must be related through $\xi_0$.

Definition. The relationship of $G(\xi_0)$ and $G(\xi_1) \sqcup G(\xi_2)$ is denoted "$\approx$" so $G(\xi_0) \approx GG(\xi_1) \sqcup G(\xi_2)$ means any relation that $\xi_0$ holds with any other schedule arises through adjustments through one of $\xi_1$ or $\xi_2$.

Definition Define addition on $J(G, P)P(U)$ as the free group generated by the elements of the set $\{G(\xi) \mid \xi \in P(U)\}$ factored by the subgroup generated by the set of relations

$$\{G(\xi_1) + G(\xi_2) - G(\xi_3) \mid G(\xi_3) \approx G(\xi_1) \sqcup G(\xi_2)\}.$$ 

Given $G(\xi)$ denote its class in $J(G, P)$ as $[G(\xi)]$.

This is a standard Grothendieck Group construction [20]. The action of the adjustments is fully consistent with sheaf structure in the sheaf of sets. This additional structure is defined in terms of sections and so it gives a group structure on the sheaf $J(G, P)$.

For the rest of this section assume that among the possible groupoids of adjustments, $G(\xi)$ is the full set of adjustments that are relevant to the $\xi$. The phrase "relevant to" is to imply that a category of groupoids has been chosen to reflect the type of adjustments that are to be studied. This means that $G$ contains all the decompositions of groupoids. If $G_1$ and $G_2$ act on $\xi$ with $g_1 \in G_1, g_2 \in G_2$, then $g_2 \circ g_1^{-1} \in G_2, g_1 \circ g_2^{-1} \in G_1$ so that $G_1 \sqcup G_2$ is a subgroupoid of the set of all adjustments on $\xi$.

The nature of $G$ for $P$ can influence the cohomology of $P$. Define $G$ as quasi-linear if for all $U \in U$ and for all $g \in G(U)$ there are $g_1$ and $g_2$ such that $g(\xi_1 + \xi_2) = g_1 \xi_1 + g_2 \xi_2$. For an arbitrary set $S \subset P(U)$ the $G$ closure of $S$ is $\bigcup_{\xi \in S} G(\xi)$.

Definition. $P$ is $G$ constrained at $U$ if there exists $V \subset U$ such that the $G$ closure of the image of the restriction map $P(U) \to P(V)$ is a proper subgroup of $P(V)$

This means that the restriction of $P(U)$ to $P(V)$ does not generate all the $\xi \in P(V)$ that can be reached by adjustments.

Proposition 6.1. If $G$ acts on $P$ quasi-linearly and $P$ is $G$ constrained at $U$ then if there is an $n$ with $\dim(Z^n(P)) \neq 0$ then $H^n(P) \neq 0$.

Proof. In the case $U = \cup U_i$ and $V = \cap U_i$ even if $\xi \in P(V)$ has an expression in terms of restrictions of $\eta_i \in P(U_i)$ there might be $g.\xi$ that cannot be expressed in terms of elements of the form $\eta_i \mid V$.

Suppose we have an $n$ cocycle in $P(V)$. Quasi-linear means we can apply adjustments to cocycles and get other cocycles:

$$\sum_{i \text{ even}} \xi_i = \sum_{i \text{ odd}} \xi_i$$
hence there exists $g$, for $i$ even and $g_j$ for $j$ odd so that

$$g \sum_{i \text{ even}} \xi_i = g \sum_{i \text{ odd}} \xi_i.$$ 

Even if the cocycle is a coboundary there are adjustments of the cocycle that cannot be expressed in terms of the original cocycle so we have a new cocycle without a corresponding coboundary - a cohomology class.

6.2 The Coordination of Changes, the Cohomology of $J(\mathcal{G}, \mathcal{P})$

The Čech cohomology construction can be applied to any group-valued sub-sheaf $\mathcal{J} \subseteq J(\mathcal{G}, \mathcal{P})$.

Starting with $C^n(\mathcal{J})$ the additive free group (Z module) is generated by the elements

$$G(\xi_{i_0, i_1, \ldots, i_n}) (U_{i_0} \cap U_{i_1} \cap \ldots \cap U_{i_n})$$

over all the non-empty intersections.

$\partial : C^n(\mathcal{J}) \to C^{n+1}(\mathcal{J})$ is defined as in section 4 with $\xi$ replaced with $\mathcal{G}(\xi)$. The resulting cohomology is related to the cohomology of $\mathcal{P}$ as follows. Given a cocycle in $\mathcal{P}$: $\xi_0 + \xi_2 + \ldots + \xi_{2k} = \xi_1 + \xi_3 + \ldots + \xi_{2k+1}$ on some $U$ and the groupoid elements act linearly or quasi-linearly then in $\mathcal{J}$

$$G(\xi_0 + \xi_2 + \ldots + \xi_{2k}) = G(\xi_1 + \xi_3 + \ldots + \xi_{2k+1}).$$

so that

$$G(\xi_0) + G(\xi_2) + \ldots + G(\xi_{2k}) = G(\xi_1) + G(\xi_3) + \ldots + G(\xi_{2k+1}).$$

However there is no rule that says these sets are the same sets. Indeed consider the following counter-example: given a 2-cocycle $\xi_0 + \xi_2 = \xi_1 + \xi_3$ on some $a \in A$ in which $Typ(a)$ has raw materials $\{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$. $\xi_0$ uses $\{e_1, e_2, e_3, e_4\}$, $\xi_2$ uses $\{e_5, e_6, e_7, e_8\}$. $\xi_1$ uses $\{e_1, e_2, e_3\}$, $\xi_3$ uses $\{e_4, e_5, e_6, e_7, e_8\}$. Adjustments can arise by substitutions in $\{e_4, e_5, e_6\}$. This leaves $G(\xi_1)$ and $G(\xi_3)$ distinct sets but $G(\xi_0)$ and $G(\xi_2)$ share adjustments by swapping around $e_4$ and $e_5$ and so $G(\xi_0) \cap G(\xi_2) \neq \emptyset$.

In the case where the map $\lambda : \mathcal{P} \to J(\mathcal{G}, \mathcal{P})$ defined by $\lambda : \xi \mapsto G(\xi)$ is linear then:

**Proposition 6.2.** If $G(\xi) \cap G(\xi') \neq \emptyset$ then $(\xi - \xi') \in ker(\lambda : \mathcal{P} \to J(\mathcal{G}, \mathcal{P}))$.

**Proof.** Let $g$ in $G(\xi) \cap G(\xi')$ then the groupoids overlap and there can be found a $g'$ so that $g' \xi = \xi'$ so $\lambda(g_0) = \lambda(\xi_0)$. \square

Apply this to $\xi = \xi_0$, $\xi' = \xi_2$. The kernel of $\lambda$ is a measures of the changes in the induced cohomology. Thus the difference in $\bar{H}^n(\mathcal{J})$ and $\bar{H}^n(\mathcal{P})$ reflects on the extent that the local variations, as indicated by cocycles that are not coboundaries, are smoothed out by adjustments.

The ability to coordinate change is an important goal for supply network managers; it is the much desired "integration". If $J(\mathcal{G}, \mathcal{P})$ is soft then every change can be accommodated throughout the supply network. The existence of useful sub-sheaves $\mathcal{P}$
of schedules with desirable properties bears significantly on questions of coordination and integration. Suppose each schedule $\xi$ in $\mathcal{S}$ can be perturbed or adjusted to one, $\gamma(\xi)$, in $\mathcal{P}$ giving a sheaf map $\gamma : \mathcal{S}(U) \to \mathcal{P}(U)$ that need not be a group homomorphism. Suppose, also, that $\gamma$ is a retract so $\gamma$ is the identity on $\mathcal{P}$.

**Definition:** $\gamma$ is $\mathcal{G}$-defined if $\forall \xi \exists g^\gamma$ such that $\gamma(\xi) = g^\gamma(\xi)$ for some $g^\gamma \in \mathcal{G}(\xi)$.

**Theorem 6.1.** If $\gamma$ is $\mathcal{G}$-defined then for all $U \in \mathcal{U}$, $J(\mathcal{G}, \mathcal{S})(U) = J(\mathcal{G}, \mathcal{P})(U)$.

**Proof.** If $\forall \xi \exists g^\gamma$ such that $\gamma(\xi) = g^\gamma$ for some $g^\gamma \in \mathcal{G}(\xi)$ then $\mathcal{G}(\gamma(\xi)) = \mathcal{G}(\xi)$ as $(g^\gamma)^{-1}(\gamma(\xi)) = \xi$ so that $J(\mathcal{G}, \mathcal{S})(U) = J(\mathcal{G}, \mathcal{P})(U)$.

If $J(\mathcal{G}, \mathcal{P})$ is soft $J(\mathcal{G}, \mathcal{S})$ itself is soft. The level of integration can be localized so that all the above considerations are only applicable to certain coordinated sub-networks $B \subset \mathcal{A}$ where $J(\mathcal{G}, \mathcal{S})(B) = J(\mathcal{G}, \mathcal{P})(B)$.

### 6.3 Adjustments and Forms; Optimization

Let $\Lambda$ be a form or forms that are desired measures of schedules. Then, as both $\mathcal{D}[\Lambda]$ and $\mathcal{G}$ are sheaves defined on $\mathcal{A}$, so therefore is their product $\mathcal{K} = (\mathcal{D}[\Lambda] \times \mathcal{G})$. Assume that for any $\phi$ in $\mathcal{D}[\Lambda]$ with $\phi(\xi)$ positive ($\phi$ has a positive norm or is in the positive cone of $\mathcal{D}[\Lambda]$) then it is desirable that $\phi(\xi)$ is as small as possible. Define a relationship sheaf $\mathfrak{G} \rightarrow \mathcal{P} \times (\mathcal{D} \times \mathcal{G})$ by $\mathfrak{G}(U)(\xi, < \phi, g >) \leftrightarrow ((\phi(U)(g(\xi(U)))) \leq (\phi(U)(\xi(U))))$.

$\mathcal{P}_{\mathfrak{G}}$ is then given by

$$\mathcal{P}_{\mathfrak{G}}(U) = \{X(\phi, g) \mid \exists \xi \in \mathcal{P}(U) \text{ and } \mathfrak{G}(\xi, (\phi, g))\}$$

where $X(\phi, g) = \{\xi \in \mathcal{P}(U) \mid \mathfrak{G}(\xi, (\phi, g))\}$

$$\mathcal{K}_{\mathfrak{G}}(U) = (\mathcal{D}[\Lambda] \times \mathcal{G})_{\mathfrak{G}}(U) = \{Y(\xi) \mid \exists \xi \in \mathcal{P}(U)\}$$

and $Y(\xi) = \{((\phi, g) \in (\mathcal{D}[\Lambda] \times \mathcal{G}))(U) \mid \phi(g, \xi) \leq \phi(\xi)\}$. This gives the existence of improved outcomes resulting from the adjustments in $\mathcal{G}$ defined for $\mathcal{P}$.

We consider two ways to give $\mathcal{P}_{\mathfrak{G}}$ a group structure. $\mathcal{P}_{\mathfrak{G}}$ is given the following group operation. For every $U \mathcal{P}_{\mathfrak{G}}(U)$ is, as usual, the free group of the elements $X(\phi, g)$ but factored as follows:

1. Factored by the relation $X(\phi_1, g_1) \dagger X(\phi_2, g_2) = X(\phi_3, g_3)$ if and only if $\xi_3 \in X(\phi_3, g_3)$ which is acted on by $g_3 \in \mathcal{G}(\xi_3)$ as $\mathcal{G}(\xi_1) \cap \mathcal{G}(\xi_2)$. This means $\xi_3$ is a mixture of $\xi_1$ and $\xi_2$ and these components can be acted on by $\mathcal{G}$ separately. $g_3$ can be separated into $g_1$ and $g_2$ so the equation will only hold if $\phi_3(\xi_3)$ is appropriated for the segments of the schedule $\xi_3$ and $\phi_1(g_1, \xi_1) + \phi_1(g_1, \xi_1) \in m(\phi_3)$.

Denote this group structure as $(\mathcal{P}_{\mathfrak{G}}, \dagger)$

2. $X(\phi_1, g_1) \boxplus X(\phi_2, g_2) = X(\phi_3, g_3)$ if and only if $g_3 = g_1 \circ g_2$ or $g_3 = g_2 \circ g_1$ and $\Vert \phi_3 \Vert \leq \Vert (\phi_1 \wedge \phi_2) \Vert$, where $\wedge$ is the lattice operation of greatest lower bound.

Denote this group structure as $(\mathcal{P}_{\mathfrak{G}}, \boxplus)$

---

11We skip the details of defining $\mathfrak{G}$ in terms of category of sheaves. This category being a Topos allows the definition of order relations and groupoid actions and so can be easily accomplished.
The second relation means that $\xi_1 \in X(\phi, g_1), i = 1, 2$ are in $X(\phi_3, g_3)$ as a $\xi_3$ for which $\phi_3(g_1 \circ g_2)(\xi_3) \leq \phi_3(\xi_4) \leq \phi_3(\xi_3)$. That is both $\xi_1$ and $\xi_2$ satisfy $\phi_3(g_1 \circ g_2)(\xi_1) \leq \phi_3(\xi_4) \leq \phi_3(\xi_3)$. That is that is a class in $\Lambda$. A cocycle equation $\sum\limits_{k=0,1, \ldots, m} X(\phi_k, g_k)$ with $k = 0, 1, \ldots, m$ that is a class in $H^m(\mathcal{P}_\phi, \mathfrak{B})$ means that there are two ways in the intersection $U = \bigcap\limits_{k=0,1, \ldots, m} U[k]$ for adjusting the schedules and locally improving their measures simultaneously. This is most clear when $\Lambda = \{\phi\}$ has only one measure. In this case the schedules

$$\xi \in X(\phi, g) = \sum\limits_{k \text{ even}} X(\phi, g_k) = \sum\limits_{k \text{ odd}} X(\phi, g_k)$$

have been subject to the adjustments of the $g_k \in G(U)$ with each $g_k$ reducing the measure $\phi$. The existence of $H^p(\mathcal{P}_\phi, \mathfrak{B})$ or $H^p(\mathcal{P}_\phi, +)$ with a high $p$ means there are schedules that can be improved over $p/2$ iterations which might correspond to distinct dimensions of adjustments. Conversely, $H^p(\mathcal{P}_\phi, \mathfrak{B})$ is zero for $p > 2$ or $3$ implies little ability to improve the measures of schedules.

### 7 Conclusion

While the concept of combinatorial systems is not new and can be found as formal models in numerous guises (see for example the collection [14]), its formulation here is intended as a algebraic structure that supports constructions of sheaves and their "Čech-like" cohomology. The intention is to gain new perspectives on systems used in many economic and business simulations. The algorithms that are manifested as schedules in combinatorial systems are "acyclic"; they contain no conditional returns ("do while", "do until" loops etc.; such constructions can be part of single elements of $W(a)$). The maps and objects, or actions and types, of a combinatorial system provide a language of transformations that give a "field" upon which schedules can be defined. If $Algo(A)$ denotes the algorithms defined on $A$ then these must correspond
to schedules. In this way we can take the perspective that if \textit{AcyAlgo}(\textit{A}) are the acyclic algorithms that can be manifested in \textit{A} we can associated them with certain sheaves:

1. \textit{AcyAlgo}(\textit{A}) \subset \textit{Comb}(\textit{A}) \xrightarrow{\text{cover}} \textit{Sh}(\textit{U}^{op}).

2. The Topos \textit{Sh}(\textit{U}^{op}) is used to add sheaves of schedule classes to \textit{Comb}(\textit{A}) by the \([\mathcal{E}, \mathcal{K}]\) construction. Can classes \(\Gamma\) of sheaves \(\mathcal{K}\) give classes of expansion of \(\textit{Comb}(\textit{A}) \rightarrow \textit{Comb}(\textit{A})[\Gamma]\)? A sequence of such classes measures the size of \(\textit{Comb}(\textit{A})\) relative to \(\textit{Sh}(\textit{U}^{op})\).

3. The totality of constructions \(\mathcal{P}_\mathcal{E}\) are added to \textit{Comb}(\textit{A}) which builds a "conceptual context" of the system \textit{A} effectively expanding what we want to say about it.

The use of a covariant cohomology to express local linear relations for both sheaves of schedules and sheaves of classes of schedules is just a start. The second part of the paper, the applications of the \([\mathcal{E}, \mathcal{K}]\) constructions indicate that much of our language about a system \textit{A} lies in the choice of the possible \(\mathcal{K}\) and is likely to have its own "grammar" given by interactions among the possible \(\mathcal{K}\). A clear example of this is \(\mathcal{K} = \mathcal{D}[\Lambda] \times \mathcal{G}\) which creates relations among relations. This is a fertile area of research for systems. How will it grow with the the size of the combinatorial systems and what does it reflect about the algorithms that can be manifested as schedules? Given that combinatorial systems are indeed good models of a wide range of natural, information and manufacturing processes what might the grammar tell us? Although it is highly abstract it is in fact part of the daily work of business systems analysts in innumerable businesses. Indeed what has driven much of the actual evolution of organizational systems has been the evolution of measures of significance.

This paper is just the start of the subject of systems that combine logical, material or data types, ideas and components to create, arrive at or "bring into being" information, ideas, devices or new materials. This paper has made it clear how the tools from algebraic topology and geometry can inform a subject that straddles pure mathematics and the very practical aspects of the way we run our society.
8 Table of Symbols

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Meaning</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>The index set of the topics in a combinatorial system</td>
<td>Section 2</td>
</tr>
<tr>
<td>$a, b, a_1, a_2,...$</td>
<td>Elements of $A$</td>
<td>section 2</td>
</tr>
<tr>
<td>$Tpc(a), Tc(b), Tpc(a_1),...$</td>
<td>Topics (categories) of elements of $A$</td>
<td>section 2</td>
</tr>
<tr>
<td>$e, e', e_1, e_2,...$</td>
<td>objects of $Tpc(a), a \in A$</td>
<td>section 2</td>
</tr>
<tr>
<td>$w, w', w_1, w_2,...$</td>
<td>maps of $Tpc(a), a \in A$</td>
<td>section 2</td>
</tr>
<tr>
<td>$A, A_1,...$</td>
<td>combinatorial systems</td>
<td>section 2</td>
</tr>
<tr>
<td>$Cat$</td>
<td>A category of small categories</td>
<td>section 2</td>
</tr>
<tr>
<td>$Typ(a)$</td>
<td>Types of a combinatorial module</td>
<td>section 2</td>
</tr>
<tr>
<td>$act(a), W(a)$</td>
<td>Actions and ring of actions of a combinatorial module</td>
<td>section 2</td>
</tr>
<tr>
<td>$f : A_1 \rightarrow A_2, Tpc(f)$</td>
<td>combinatorial system morphism</td>
<td>section 2.1</td>
</tr>
<tr>
<td>$Comb$</td>
<td>the category of combinatorial systems</td>
<td>section 2.1</td>
</tr>
<tr>
<td>$CombMod$</td>
<td>the category of combinatorial modules</td>
<td>section 2.1</td>
</tr>
<tr>
<td>$\xi, \xi', \xi_1,...\eta, \zeta$</td>
<td>Schedules</td>
<td>section 2.1</td>
</tr>
<tr>
<td>$U, U_1 U$</td>
<td>subsets of $A$, covering of $A$</td>
<td>section 2.1</td>
</tr>
<tr>
<td>$Sch$</td>
<td>The sheaf of schedules</td>
<td>section 2.1</td>
</tr>
<tr>
<td>$[\mathcal{L}, k]$</td>
<td>The relative sheaf construction</td>
<td>section 3.1</td>
</tr>
<tr>
<td>$\mathcal{P}_L, k_L$</td>
<td>Class sheaves of $\mathcal{L}$</td>
<td>section 3.1</td>
</tr>
<tr>
<td>$\Lambda, D[\Lambda]$</td>
<td>Sets of forms, the sheaf of $\Lambda$ forms</td>
<td>section 3.1</td>
</tr>
<tr>
<td>$\mathcal{E}, \mathcal{P}<em>\mathcal{E}, D[\Lambda]</em>{\mathcal{E}}$</td>
<td>Class sheaves for forms</td>
<td>section 3.2</td>
</tr>
<tr>
<td>$\mathcal{G}, s, t : \mathcal{G} \rightleftharpoons \mathcal{P}$</td>
<td>Groupoid fibration over $\mathcal{P}$</td>
<td>section 3.3</td>
</tr>
<tr>
<td>$C^n(\mathcal{P}, U), C^n$</td>
<td>Cohomology for the sheaf $\mathcal{P}$ and cover $U$</td>
<td>section 4</td>
</tr>
<tr>
<td>$\partial^n : C^n \rightarrow C^{n+1}$</td>
<td>The cochain homomorphism</td>
<td>section 4</td>
</tr>
<tr>
<td>$Z^n, Z^n(\mathcal{P}, U), B^n, B^n(\mathcal{P}, U)$</td>
<td>The n-th cocycle and coboundary groups</td>
<td>section 4</td>
</tr>
<tr>
<td>$H^n, H^n(\mathcal{P}, U)$</td>
<td>The n-th cohomology group</td>
<td>section 4</td>
</tr>
<tr>
<td>$Inv, Scn(\mathcal{A})$</td>
<td>The &quot;inventory&quot; sheaf and sheaf of scenarios of $\mathcal{A}$</td>
<td>section 5.1</td>
</tr>
<tr>
<td>$P_{\text{inv}}, Scn_{\text{inv}}$</td>
<td>Class sheaves of schedules satisfying scenarios</td>
<td>section 5.1</td>
</tr>
<tr>
<td>$\mathcal{S}ch(\mathcal{S}(x, q)) \equiv \mathcal{S}ch(\mathcal{S}(x, q))$</td>
<td>Sheaf of schedules fulfilling $\mathcal{S}(x, q)$</td>
<td>section 5.2</td>
</tr>
<tr>
<td>$J(\mathcal{G}, \mathcal{P})$</td>
<td>The class sheaf functor of the groupoid $\mathcal{G} \rightleftharpoons \mathcal{P}$</td>
<td>section 6.1</td>
</tr>
<tr>
<td>$G(\xi_3) \equiv G(\xi_1) \sqcup G(\xi_2)$</td>
<td>Groupoid actions on $\xi_3$ decompose into actions on $\xi_1, \xi_2$</td>
<td>section 6.1</td>
</tr>
<tr>
<td>$\mathcal{G} \Rightarrow \mathcal{P} \times (D \times \mathcal{G})$</td>
<td>&quot;Optimizing&quot; relationship sheaf</td>
<td>section 6.3</td>
</tr>
</tbody>
</table>

ANDREW MACFARLANE  
Integrated Expertise Research  
3a Jeta Gardens  
27 Clarendon Avenue  
Bethania  
Queensland  
Australia 4205.  
Email: combinatorial.schema@gmail.com.

References


