Universität Zürich
Institut für Informatik
Herbstsemester 2015 und Frühjahressemester 2016

Bachelor of Arts in Sozialwissenschaften
Nebenfach Informatik (30 ECTS)

Verantwortlicher Professor
Prof. Dr. Lorenz M. Hilty

Facharbeit

pymfa2:
Ein Werkzeug zur Analyse von Materialflüssen in Python 2.7

7. Februar 2016

Autor

Rolf Badat
geboren in Lachen SZ
Matrikelnummer: 07-743-289

Freihofstrasse 33
8048 Zürich

rolfbadat@bluewin.ch
Mobile: 078 618 41 23
Politikwissenschaft (Hauptfach)
Informatik (kleines Nebenfach)
Umweltwissenschaften (kleines Nebenfach)

Inhaltsverzeichnis

1. Einleitung

Viele verschiedene Materialien kommen in ganz unterschiedlichen Produkten vor. Mit einer Materialfluss-Analyse (MFA) versucht man, den Fluss der Materialien über den Lebenszyklus der Produkte hinaus, bis sie in Senken (zum Beispiel Boden, Wasser, Luft) anfallen, nachzuzeichnen oder vorauszusagen. Mit dem Wissen über diese Materialflüsse können die von ihnen verursachten Risiken für die Natur – sei es in den Senken oder auf dem Weg dorthin – erkannt werden und mögliches Potential zur Risikoverminderung im Materialfluss lokalisiert werden. Auch die Höhe dieses Potentials kann mittels einer MFA genauer eingeschätzt werden. Dies ermöglicht es, Materialflüsse nach Möglichkeit in der Art zu ändern, dass sie ein kleineres Risiko für die Umwelt darstellen.
Um diese Materialflüsse zu modellieren, verwendet man Graphen. Unternehmen, Kunden, Recyclingstellen oder Senken (wie Boden oder Wasser) werden als Knoten dargestellt. Die Verbindungen repräsentieren die Materialflüsse zwischen den Knoten und geben an, wie viel des Materials wohin weitergeleitet wird.
Die vorliegende Arbeit ist eine Software, die zur Berechnung der Materialflüsse von Materialflussmodellen dient, die zugleich dynamisch und probabilistisch sind. Ausgehend vom Programm *pymfa* (Alexandru 2013), welches deterministische Modelle berechnet und relativ einfach auch ohne Programmierkenntnisse bedient werden kann, wurde eine Verbindung zum Programm *dpmfa* (Bornhöft et al. 2015) geschaffen, welches probabilistische Modelle rechnen kann aber noch keine Schnittstelle zur Benutzerin bietet.
In diesem schriftlichen Bericht wird zunächst auf die verwendete Drittsoftware eingegangen und dann die Implementation der vorliegenden Software erläutert. Weiter folgt ein Ausblick auf mögliche zukünftige Erweiterungen der Software und zum Schluss wird ein Fazit gezogen.

2. Software Dritter

Die vorliegende Software baut auf zwei bereits existierenden dynamischen MFA Softwarelösungen auf: Das Programm *pymfa* (Alexandru 2013) rechnet deterministische Modelle und dient als Grundlage. Um probabilistische Modelle rechnen zu können, werden die Berechnungen aber auf das Framework *dpmfa* (Bornhöft et al. 2015) ausgelagert.

2.1. pymfa

Dieses Tool rechnet Materialfluss-Modelle anhand von Daten, die via csv-Datei eingelesen werden. Die Idee für diese Datenübergabe wurde für die vorliegende Software übernommen. Da *pymfa* aber nur deterministische Modelle berechnet und die Daten für die Berechnung eines probabilistischen Systems im *dpmfa-Simulato*r eine andere Struktur erfordern, wurde weiter nichts von *pymfa* nach *pymfa2* übernommen.

2.2. dpmfa

Der *dpmfa-Simulator* rechnet dynamische probabilistische Modelle. Mittels Bayes'scher Modellierung wird unvollständiges Wissen über verschiedene Parameter im Modell berücksichtigt. Mit einer Monte-Carlo-Simulation werden in den verschiedenen Durchläufen Werte für die einzelnen Parameter gezogen. Die damit berechneten Mittelflüsse werden für jede Periode in jedem Durchlauf gespeichert und können nach Bedarf abgerufen werden.
Es stehen die Knotentypen `FlowCompartment`, `Sink` und `Stock` zur Verfügung. Im `FlowCompartment` werden Materialflüsse unmittelbar mit dem Eintreffen weiter an die Zielknoten geleitet. In einer Senke (`Sink`) werden die Materialflüsse nicht weitergeleitet. Sie sammeln sich in diesem Knoten an. Im `Stock` können die eintreffenden Materialflüsse verzögert und mittels Release Strategie über mehrere Perioden verteilt an die Zielknoten weitergeleitet werden.
Die in *dpmfa* vorgefundene Implementation erlaubt es nicht, für jede Periode eigene Parameter für die Transfers und Release Strategien festzulegen. Auch ist es vorgesehen, dass die Materialflüsse aus einem `Stock` zu den Zielknoten alle mit derselben Release Strategie auskommen müssen. Da sich die Parameter aber abhängig von der Zeit ändern können, wurden für die Eingliederung in pymfa2 die neuen Klassen `TDRStock`, `PeriodDefinedRelease` und `PeriodDefinedTransfer` kreiert, die Klasse `FlowCompartment` und die Berechnungen über alle Module des dpmfa-Simulators angepasst. Die geänderten Stellen sind im Code als geändert gekennzeichnet. Es wurde darauf geachtet, dass der dpmfa-Simulator nichts an seiner Funktionalität verliert. Noch immer könnten die alten Klassen benutzt werden, würde dies die Dateneingabe via csv-datei erlauben.

3. Implementation

Die Software pymfa2 ist gänzlich in Python 2.7 geschrieben. Sie bedient sich der externen Bibliotheken *numpy*, *scipy* und *matplotlib*. Um sie ausführen zu können, müssen Python 2.7 und die erwähnten Bibliotheken auf dem Rechner installiert sein. Ist dies der Fall, kann das Programm mit einer Konsole (zum Beispiel Windows Eingabeaufforderung) ausgeführt werden. Hierzu setzt man in der Konsole den Pfad auf den Ordner 'pymfa2' und gibt den Dateinamen 'runner.py' sowie zwei Argumente ein. Das erste Argument ist der Pfad und der Name zur comma-separated-values-Datei (csv-Datei), welche als Datenquelle für die Berechnungen dient. Das zweite Argument besteht aus Pfad und Dateiname für die Sicherung der Resultate in einer csv-Datei. Sind diese Eingaben gemacht, kann das Programm ausgeführt werden.

3.1. Dateneingabe

Das Modell, welches mit dem Programm gerechnet werden soll, muss mittels einer csv-Datei eingespeist werden. Dieses Format bietet sich deshalb besonders gut an, da es plattformunabhängig ist und sich Daten aus Microsoft Excel oder anderen Tabellenkalkulationsprogrammen sehr einfach als csv-Datei exportieren lassen.

Das Programm liest nur diejenigen Zeilen, die in der ersten Spalte einen Eintrag haben. Dies ermöglicht es der Benutzerin, Kommentare in der csv-Datei einzubinden, sofern jeweils die erste Spalte leer gelassen wird. Das Weiteren muss die csv-Datei wie folgt aufgebaut sein:

- In der ersten gelesenen Zeile muss angegeben werden, wie viele Läufe in der Monte-Carlo-Simulation simuliert werden sollen. Trage 'Runs:' in die erste Spalte und die Anzahl in die nächste Spalte ein.
- In der zweiten gelesenen Zeile muss angegeben werden, wie viele Perioden (Jahre) für die Berechnungen berücksichtigt werden sollen. Trage 'Periods:' in die erste Spalte und die Anzahl in die nächste Spalte ein. Wird die Zahl '0' eingegeben oder die Zelle leer gelassen, wird die Anzahl Jahreszahlen aus der weiter unten folgenden Tabelle übernommen. Es müssen mindestens zwei Perioden berechnet werden.

- In der dritten gelesenen Zeile muss angegeben werden, ob ein Medianwert mitberechnet werden soll. Trage 'Median:' in die erste Spalte und nichts, '0', 'n', 'no' oder 'none', respektive '1', 'y' oder 'yes' in die nächste Spalte ein. Im Gegensatz zu den Medianwerten werden die Mittelwerte immer mitberechnet
- In der vierten gelesenen Zeile muss angegeben werden, welche Perzentile berechnet werden sollen. Trage 'Percentiles:' in die erste Spalte und die Perzentile (Zahlen zwischen 0 und 100, getrennt durch '|') oder nichts in die nächste Spalte ein.
- In der fünften gelesenen Zeile muss angegeben werden, für welche Knoten aus der weiter unten folgenden Tabelle Plots von Materialzu-/-abflüssen und Inventare über die ganze simulierte Zeitspanne erstellt und gespeichert werden sollen. Trage 'Plots:' in die erste Spalte und die Knotennamen (getrennt durch '|') in die nächste Spalte. Wird nichts oder die Zahl '0' eingegeben, werden keine Plots erstellt. Wird die Zahl '1' eingegeben, werden für alle Knoten ausser den 'conversions' Plots erstellt.
- Als nächstes folgt die Titelzeile für die Datentabelle. Von links nach rechts sollen folgende Einträge gemacht werden: 'Transfer Type', 'Source Node', 'Source Material', 'Source Unit', 'Target Node', 'Target Material', 'Target Unit', 'Description'. In die weiteren Spalten sind die Jahreszahlen aufsteigend von links nach rechts einzutragen.

Nun folgt die Eingabe der Tabelle unter die vorhin eingegebene Titelzeile. Hier stellt jede Zeile einen Transfer von Material zwischen zwei Knoten dar. Unter 'Transfer Type' wird die Art des Transfers eingetragen. Es ist aus folgenden Transfertypen zu wählen:

- *Inflow*: Bezeichnet einen Materialtransfer von aussen in das untersuchte System. Die Tabelle muss immer mit einem Transfer dieses Typs beginnen.
  Rate: Bezeichnet einen Transfer zwischen einem Quellknoten und einem oder mehreren Zielknoten, ohne dass dabei das Material umgewandelt wird.
- *Conversion*: Dieser Transfer erlaubt es, ein Material in eine andere Masseinheit umzuwandeln. Zum Beispiel werden aus 1000 Stück Laptops 3000 Kg gemischtes Material.
- *Fraction*: Dieser Transfer erlaubt es, Material in andere Materialien aufzusplitten. Zum Beispiel werden aus 3000 Kg gemischtem Material 1200 Kg Plastik und 1800 Kg Metall.
- *Delay*: Dieser Transfer kann Material verzögert an die Zielknoten weitergegeben.

Weiter werden die Namen der Knoten und die Materialart und -einheit eingetragen. Für eine 'Fraction' muss keine Zieleinheit eingetragen werden, da diese sich nicht ändert. Für die Transfertypen 'Rate' und 'Delay' muss kein Zielmaterial oder keine Zieleinheit eingetragen werden, da sich diese ebenfalls nicht ändern.
Unter 'Description' kann ein kurzer Text erfasst werden, der den Transfer genauer beschreibt.
Unter den Jahreszahlen werden für jede Zeile und jedes Jahr die Angaben zu den Transferkoeffizienten (TK) gemacht. Ein TK muss fixiert ('fix'), stochastisch ('stoch') oder zufällig ('rand') gezogen werden. Wird 'fix' gewählt muss zusätzlich der TK angegeben werden. Wird 'rand' gewählt muss zusätzlich eine Liste mit durch Komma getrennte TKs angegeben werden. Wird 'stoch' gewählt muss zusätzlich angegeben werden, unter welcher Wahrscheinlichkeitsdichtefunktion die TKs in jedem einzelnen Durchlauf der Monte-Carlo-Simulation gezogen werden sollen. Zur Auswahl stehen die uniforme Verteilung ('uniform'), die Normalverteilung ('norm') und die Dreiecksverteilung ('triangular'). Wird 'uniform' gewählt, muss eine Ober- und Untergrenze für die TKs angegeben werden. Wird 'norm' gewählt muss der Mittelwert der TKs und die Standardabweichung angegeben werden. Wird 'triangular' gewählt, muss zusätzlich der kleinst mögliche TK, der am wahrscheinlichsten eintretende TK und der höchst mögliche TK angegeben werden.
Da die Summe der TKs eines Knoten nicht immer 1 sein wird, müssen die TKs auf 1 normalisiert werden. Ansonsten würden im Modell plötzlich Materialien verschwinden oder entstehen. Durch eine unterschiedliche Priorisierung der einzelnen ausgehenden TKs eines Knoten, werden Werte mit höherer Priorität von der Normalisierung ausgeschlossen.
Einträge unter den Jahreszahlen können also wie folgt aussehen:
'fix|0.4|1', 'stoch|triangular|0.24, 0.3, 0.36|1', 'rand|0.2, 0.22, 0.23|1', stoch|normal|0.7, 0.1|2
Für die Transfers des Typs 'Delay' müssen zusätzlich Informationen zur Verzögerung des Materialflusses erfasst werden. Das Material eines Knoten wird solange verzögert weitergegeben, bis der ganze Materialeingang aus einer bestimmten Periode an den Zielknoten weitergeleitet wurde. Es kann aus den Verzögerungsfunktionen 'fix', 'rand', 'list' und 'weibull' gewählt werden. Wird 'fix' gewählt, wird der Materialfluss für die folgenden Perioden mit einer festen Rate weitergeleitet. Wird 'rand' oder 'list' gewählt, wird für jede Periode die Rate aus einer Liste von Raten ausgewählt – für 'rand' zufällig und für 'list' der Reihe nach. Wird 'weibull' gewählt, werden die Raten für die Perioden nach einer

Weibullfunktion ausgewählt. Diese Funktion wird von scipy.stats[1] übernommen und kann wahlweise mit zwei oder drei Parametern bestimmt werden. Danach muss noch eingegeben werden, um wie viele Perioden der verzögerte Materialfluss verzögert werden soll. Eine Eingaben unter den Jahreszahlen für den Transfertypen 'Delay' können wie folgt aussehen:
'fix|1|1|weibull|1, 3, 1|0', fix|0.5|2|weibull|1, 3, 1|2', stoch|uniform|0.5, 0.7|1|list|0.3, 0.35, 0.4|1

Wurden alle Angaben wie oben ausgeführt gemacht, ist ein Modell nun vollständig bestimmt. Ein Beispiel-Modell findet man im Anhang unter 7.1. Beispiel Dateneingabe.

3.2. Datenausgabe

Die Datenausgabe erfolgt als csv-Datei und es werden – wenn gewünscht – Plots der Zu- und Abflüsse sowie der Inventare der verschiedenen Knoten erstellt (siehe Abbildung 1). Sie werden in einem Ordner 'plots' am gleichen Ort wie die csv-Datei abgelegt.
Die csv-Datei mit den Resultaten zeigt nebst den Zahlen zu den Materialflüssen zwischen den einzelnen Knoten auch die Inventare von Senken und anderen Knoten, die ihr Material nur verzögert weiterleiten.
Falls in der Dateneingabe die Berechnung von Medianwerten und Perzentilen verlangt wurde, werden diese in der csv-Datei und in den Plots nun angegeben.

3.3. Implementation

Hier werden die einzelnen Code-Module genauer erläutert.

- *runner.py*: Dieses Modul ist der Startpunkt der Berechnungen und muss für die Verwendung des Programms ausgeführt werden. Das Modul wurde von Alexandru (2013) (da als pymfa-cli.py) mit leichten Abänderungen übernommen.
- *importer.py:* Hier werden die Modellspezifikationen aus der csv-Eingabedatei Zeile für Zeile und Spalte für Spalte überprüft. Danach werden die Daten für einzelne Transfers zwischen Knoten erfasst und in derart gespeichert, dass Transfers und verzögerte Materialflüsse nun zum jeweiligen Quellknoten gehören. Diese Datenstruktur wird von den in *linker.py* spezifizierten Klassen (`Inflowdata`, `RateData`, `DelayData` und `SinkData`) zur Verfügung gestellt.

1http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.stats.dweibull.html

- *linker.py*: Dieses Modul dient dazu, die gesammelten Daten so aufzubereiten, dass sie dem dpmfa-Simulator für die Berechnungen übergeben werden können. Ist dieser Schritt vollzogen, werden die Daten dem dpmfa-Simulator zur Modellberechnung übergeben.
- *adjusted_functions.py*: Dieses Modul liefert Funktionen, die für die Berechnung der TKs und der verzögerten Materialflüsse verwendet werden können.
- *exporter.py*: Hier wird das vom dpmfa-Simulator berechnete Modell in eine csv-Datei geschrieben, welche dann als Resultat am gewünschten Ort gespeichert wird. Sollen zusätzlich Plots von einzelnen Knoten erstellt werden, wird dies ebenfalls hier getan.
- *dpmfa-simulator.components.py*: Dieses Modul bietet die Klassen für verschiedene Knoten, Transfers und Verzögerungsstrategien, die für die Berechnung des Modells notwendig sind. In der vorliegenden Software werden aber nur die neu implementierten Kompartimente `TDRStock` als Stock-Klasse, `PeriodDefinedRelease` als Release-Klasse und `PeriodDefinedTransfer` als Transfer-Klasse verwendet. Sie ermöglichen eine Anpassung der Parameter auch über die Zeit und pro Zielknoten.
- *dpmfa-simulator.model.py*: Hier wird aus den einzelnen Knoten und verschiedenen Transfers das zu berechnende Modell erstellt.
- *dpmfa-simulator.simulator.py*: Hier wird die Monte-Carlo-Simulation für das Modell gerechnet. Für jeden Durchlauf werden Materialflüsse berechnet und gespeichert.

4. Mögliche Erweiterung der Applikation

Die hier vorliegende Applikation kann in vieler Hinsicht noch verbessert oder ergänzt werden. Zum einen wäre es möglich, pymfa2 als Online-Tool verfügbar zu machen, und ähnliche Visualisierungsmöglichkeiten wie sie im Vorgänger pymfa vorhanden sind, zu implementieren.

Bei der bereits vorhandenen Option, Plots über die Zeit für Knoten darzustellen, könnten auch noch weitere Arten von Plots generiert werden: So zum Beispiel ein Plot pro Abfluss von einem Quellknoten zum Zielknoten.

Die Verzögerungsfunktionen für Transfers sind jetzt noch deterministisch. Um auch hier einen

Unsicherheitsfaktor für die einzelnen Werte einbauen zu können, müsste die Spezifikation dieses Faktors durch die Benutzerin in der csv-Datei erfolgen. Da meiner Meinung nach die Eingabe der einzelnen Parameter unter den Jahreszahlen jetzt schon ziemlich unübersichtlich ist, ist eine Umstrukturierung dieser Dateneingabe zu empfehlen.
Vom dpmfa-Simulator können zurzeit nicht alle Klassen von Inflow-, Knoten- und Transferarten genutzt werden, da diese zusätzliche Auswahlmöglichkeit die Eingabe mittels csv-Datei zusätzlich unübersichtlich gemacht hätte. So steht zum Beispiel eine bereits durch Bornhöft et al. (2015) implementierte 'Inflowfunktion über alle Perioden' nicht zur Verfügung.

5. Fazit

Die hier vorliegende Software wurde zunächst als Erweiterung des Tools *pymfa* gedacht. Sie sollte es ermöglichen, mit dem Programm auch probabilistische Modelle zu berechnen. Nachdem aber abgesehen von Teilen aus dem Modul pymfa-cli.py und der Idee der Dateneingabe via einer csv-Datei nichts weiter übernommen wurde, kann man eigentlich nicht mehr von einer Erweiterung von *pymfa* sprechen. Viel mehr ist es ein neues Programm.
Für den Teil der Berechnung des probabilistischen Modells wurde das Framework *dpmfa* verwendet. Um die verschiedenen Parameter des Modells auch über die Zeit und für alle aus einem Knoten ausgehenden Materialabflüsse anpassen zu können, wurden verschiedene neue Klassen kreiert und die Berechnungen in *dpmfa* angepasst.
Den vorgesehenen Aufwand wurde durch die nötig gewordenen Anpassungen in *dpmfa* um ein Vielfaches überschritten. So blieb leider keine Zeit mehr, das Programm so herzurichten, dass es als Onlinelösung auch via Internet bedient werden kann.

6. Quellenangabe

Alexandru, Carol (2013): *pymfa – A Tool for Performing Material Flow Analyses in Pathon 3.* Universität Zürich.

Bornhöft, Nikolaus A., Hilty, Lorenz M., Nowack, Bernd und Sun, Tianyin (2015): *A Dynamic Probabilistic Material Flow Modeling Method.* Preprint. Empa St. Gallen und Universität Zürich. Software online erhältlich unter: https://pypi.python.org/pypi/dpmfa-simulator [Stand 06.02.2016].

7. Anhang

7.1. Beispiel Dateneingabe

Runs:	500									
Periods:										
Median:	yes									
Percentiles:	10|90									
Plots:	Dumping|Incineration									
Transfer Type	**Source Node**	**Source Material**	**Source Unit**	**Target Node**	**Target Material**	**Targer Unit**	**Description**	**1989**	**1990**	**1991**
Inflow				DirectSale	Laptops	Pieces	...	stoch|normal|90000, 6000	stoch|normal|95000, 5000	stoch|normal|125000, 4000
Inflow				OnlineSale	Laptops	Pieces	...	stoch|triangular|29000, 30000, 31000	stoch|triangular|56000, 60000, 64000	stoch|triangular|99500, 100000, 100500
Rate	DirectSale	Laptops	Pieces	Business Use			...	stoch|triangular|0.15, 0.2, 0.25|1	stoch|triangular|0.3, 0.35, 0.4|1	stoch|triangular|0.35, 0.4, 0.45|1
Rate	DirectSale	Laptops	Pieces	Private Use			...	stoch|normal|0.5, 0.1|2	stoch|normal|0.5, 0.1|2	stoch|normal|0.5, 0.1|2
Rate	DirectSale	Laptops	Pieces	Public Use			...	stoch|uniform|0.1, 0.3|1	stoch|uniform|0.1, 0.3|1	stoch|uniform|0.05, 0.15|1
Rate	OnlineSale	Laptops	Pieces	Business Use			...	rand|0.45, 0.48, 0.5, 0.52, 0.55|1	rand|0.45, 0.48, 0.5, 0.52, 0.55|1	rand|0.45, 0.48, 0.5, 0.52, 0.55|1
Rate	OnlineSale	Laptops	Pieces	Private Use			...	fix|0.4|2	fix|0.35|2	fix|0.4|2
Rate	OnlineSale	Laptops	Pieces	Public Use			...	stoch|normal|0.1, 0.02|3	stoch|normal|0.15, 0.02|3	stoch|normal|0.1, 0.02|3
Delay	Business Use	Laptops	Pieces	Business Decommissioning			...	fix|1|1|weibull|1, 5, 1|0	fix|1|1|weibull|1, 4, 1|0	fix|1|1|weibull|1, 3, 1|0
Delay	Private Use	Laptops	Pieces	Private Decommissioning			...	fix|1|1|weibull|1, 5|0	fix|1|1|weibull|1, 4|0	fix|1|1|weibull|1, 3|0
Delay	Public Use	Laptops	Pieces	Public Decommissioning			...	fix|1|1|weibull|1, 3|0	fix|1|1|weibull|1, 3|0	fix|1|1|weibull|1, 3|0
Rate	Business Decommissioning	Laptops	Pieces	Decommissioning			...	fix|1|1	fix|1|1	fix|1|1
Rate	Private Decommissioning	Laptops	Pieces	Decommissioning			...	fix|1|1	fix|1|1	fix|1|1
Rate	Public Decommissioning	Laptops	Pieces	Decommissioning			...	fix|1|1	fix|1|1	fix|1|1
Conversion	Decommissioning	Laptops	Pieces	Preprocessing	Mixed Materials	Kg	...	stoch|triangular|2.7, 2.9, 3.1|1	stoch|triangular|2.6, 2.8, 3|1	stoch|triangular|2.7, 2.8, 2.9|1
Fraction	Preprocessing	Mixed Materials	Kg	Plastic Processing	Plastic		...	rand|0.78, 0.79, 0.8|1	rand|0.78, 0.79, 0.8|1	rand|0.78, 0.79, 0.8|1
Fraction	Preprocessing	Mixed Materials	Kg	Meta Processing	Metal		...	stoch|uniform|0.1, 0.2|1	stoch|uniform|0.1, 0.2|1	stoch|uniform|0.1, 0.2|1
Fraction	Preprocessing	Mixed Materials	Kg	RecyclingParts	Parts		...	stoch|uniform|0.04, 0.06|1	stoch|uniform|0.04, 0.06|1	stoch|uniform|0.04, 0.06|1
Rate	Plastic Processing	Plastic	Kg	Incineration			...	stoch|normal|0.95, 0.1|1	stoch|normal|0.95, 0.1|1	stoch|normal|0.95, 0.1|1
Rate	Plastic Processing	Plastic	Kg	Dumping			...	stoch|normal|0.05, 0.01|1	stoch|normal|0.05, 0.01|1	stoch|normal|0.05, 0.01|1
Rate	Meta Processing	Metal	Kg	Incineration			...	stoch|normal|0.3, 0.05|1	stoch|normal|0.3, 0.05|1	stoch|normal|0.3, 0.05|1
Rate	Meta Processing	Metal	Kg	Dumping			...	stoch|normal|0.7, 0.1|1	stoch|normal|0.7, 0.1|1	stoch|normal|0.7, 0.1|1