# Analyzing the Performance of Bike-Sharing Systems

Maurice ter Beek (Formal Methods and Tools, ISTI–CNR, Pisa)

Joint work with Stefania Gnesi, Mieke Massink, Diego Latella (FMT, ISTI–CNR), Axel Legay (Inria Rennes, France), Alberto Lluch Lafuente (DTU, Denmark), and Andrea Vandin (IMT Lucca, Italy)

QUANTICOL meeting FMT – PisaMo – **Bici**nCittà
ISTI–CNR, Pisa
20 january 2017

quanticol

COOPERATION

# Outline

- Bike-sharing systems and bikes seen as product lines (configurable systems)
  - Multi-objective optimization with CLAFER
- Performance analysis of bike-sharing system (re)configurations and behavior
  - Mean field model checking with FLYFAST
  - Statistical model checking with QFLAN

# Variants of bike-sharing systems/bikes



Deciding a bike-sharing system (BSS) for a city poses many questions

- How many/what kind of bikes (features like light, basket, engine)?
- How many/what kind of stations (capacity), where to place them?
- How to avoid stations being completely full or empty for periods?
- Which BSS features (like maintenance, antitheft, smart services)?
- With or without dynamic redistribution of bikes? And what kind?
- Incentives (rewards) for users bringing bikes to less popular stations?
- Subscription costs and charging policy (like credit card or keycard)?

How to evaluate the numerous options, costs/benefits, improvements and changes in a systematic way? (i.e. performance, behavior)
Can we develop suitable decision support tools?

# Product line (PL)



Product: a valid combination (configuration) of features 

Product line or family: a set of valid feature combinations of a domain

Software Product Line Engineering (SPLE): Develop and maintain a (software) PL using a shared architecture or platform (commonalities) and mass customization (variabilities) to serve, e.g., different markets, thus allowing for (software) reuse

Aim: Maximize commonalities whilst minimizing cost of variations (i.e. of individual products)

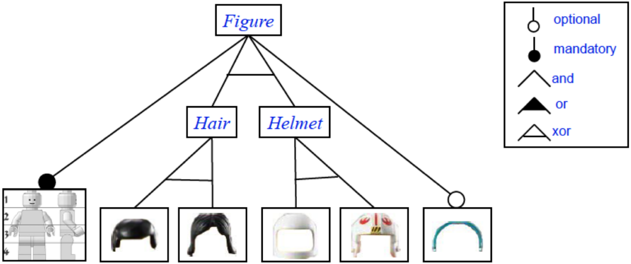
# Variability modelling

(Lego example by S. Apel, University of Passau, Germany)



Variability in terms of features and constraints:

- stakeholder visible pieces of functionality of a system . . .
- . . . which may be optional and/or may have alternatives
- only specific feature combinations lead to valid products




$Visor \Rightarrow Helmet$

Feature model: compact representation of all valid products of a PL

# Scalability is a major issue!

(slide by C. Kästner, Carnegie Mellon University, USA)



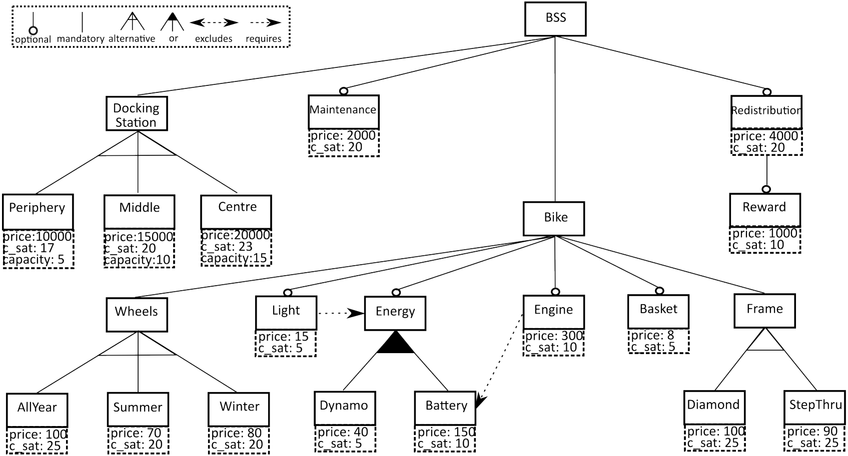
33 optional, independent features

a unique product for every

person on this planet

# Attributed feature model of a BSS

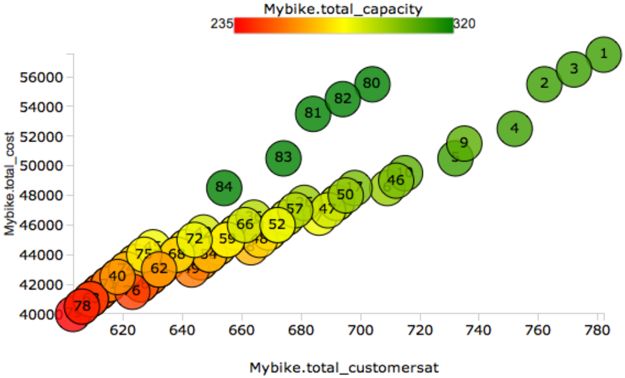





*product*: subset of *features* satisfying all variability constraints

$$cost(product) = \sum \{\, cost(feature) \mid feature \in product \,\}$$

ClaferMoo Visualizer: compare different system configurations (product variants) w.r.t. various quality dimensions, select the most desirable variant, possibly by resolving trade-offs, and understand the impact of reconfigurations on a variant's quality dimensions




(minimizing cost whilst maximizing customer satisfaction and capacity)

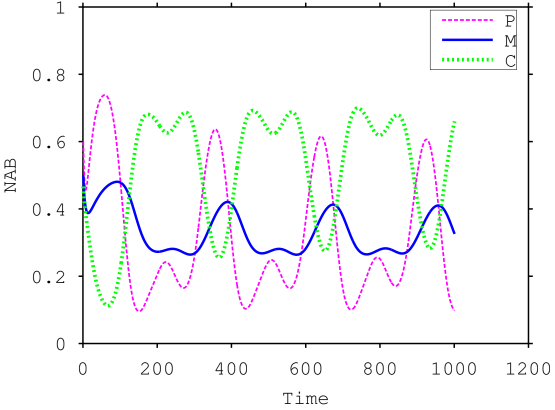
# Performance analysis BSS configuration



Assess performance of a BSS with mean field model checker FlyFast
(developed by Latella & Massink together with Michele Loreti (UNIFI))

Configuration 1 330 stations: 100 of type **P**eriphery with capacity 5
150 of type **M**iddle with capacity 10
80 of type **C**entre with capacity 15

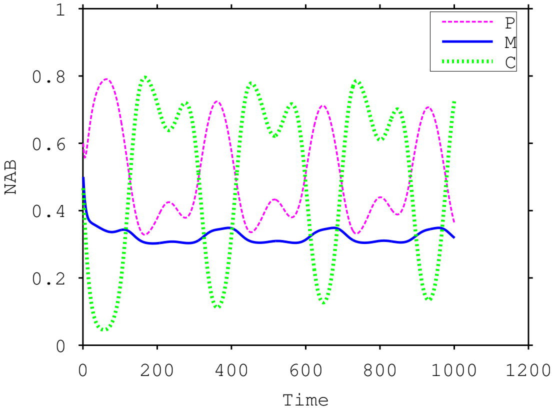
average filling degree of docking stations over time



# Performance analysis of reconfiguration

quanticol
www.quanticol.eu

Configuration 2 390 stations: 200 of type **P**eriphery with capacity 5
150 of type **M**iddle with capacity 10
40 of type **C**entre with capacity 15

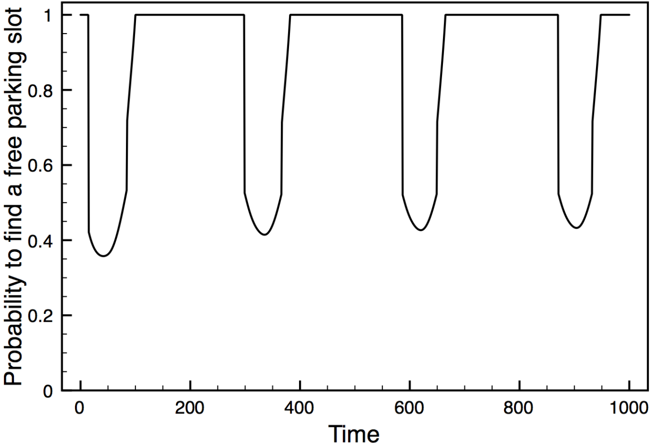



While P-type stations are in general less empty in configuration 2, filling degree of C-type stations fluctuates more in configuration 2

# Analysis of individual docking stations

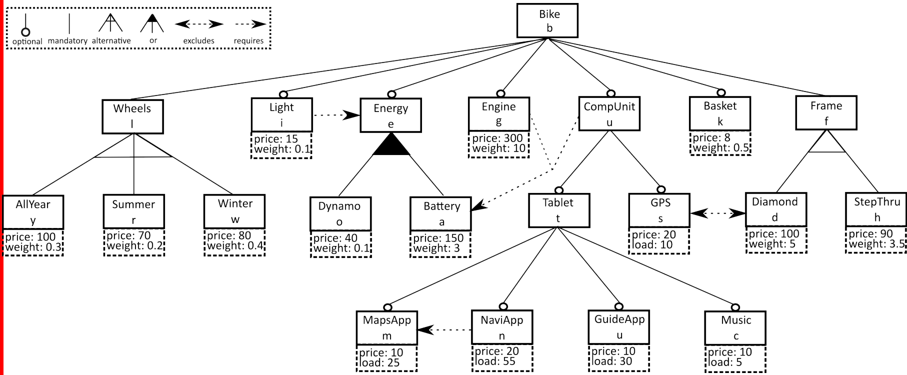


What's the probability that, within 30 mins, a peripheral station gets full, but then, with high probability, has a free slot within 6 mins, or doesn't get full, but then, with low probability, gets full within 6 mins?



# Bike product line: structural constraints



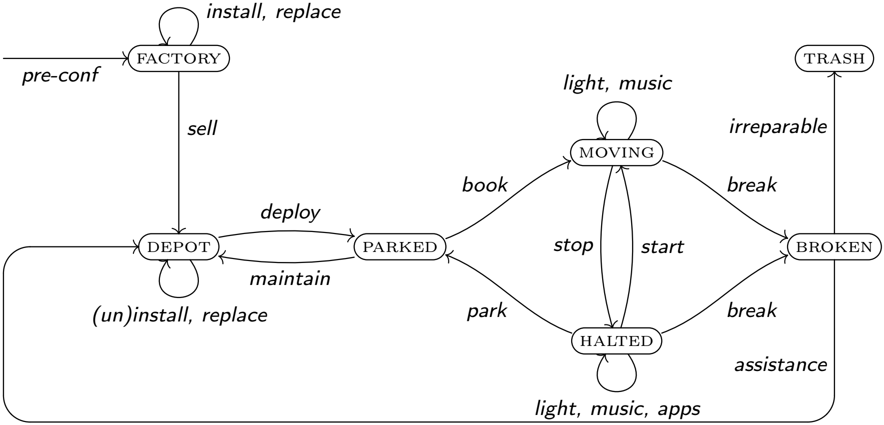



## Additional feature constraints

- C1 $\sum_{f \in \mathcal{P}_{\mathcal{F}}} price(f) \leq 600$: a bike may cost at most 600 €
- C2 $\sum_{f \in \mathcal{P}_{\mathcal{F}}} weight(f) \leq 15$: a bike may weigh up to 15 kg
- C3 $\sum_{f \in \mathcal{P}_{\mathcal{F}}} load(f) \leq 100\%$: a bike's total computational load may not exceed 100%

# Bike product line: behavioral constraints






## Additional action constraints

C4 $do(sell) \rightarrow \sum_{f \in \mathcal{P}_{\mathcal{F}}} price(f) \geq 250$

C5 $do(irreparable) \rightarrow \sum_{f \in \mathcal{P}_{\mathcal{F}}} price(f) \leq 400$

Assess performance of Bike PL with statistical model checker QFLAN
(developed by Vandin together with Ter Beek, Lluch Lafuente & Legay)

- $P_1$ Average price, weight or load of a bike
- $P_2$ For each feature, the probability to be installed
- $P_3$ The probability for a bike to be disposed of (irreparable)

When evaluated at a bike's first deployment (for two versions of C1 + C2):

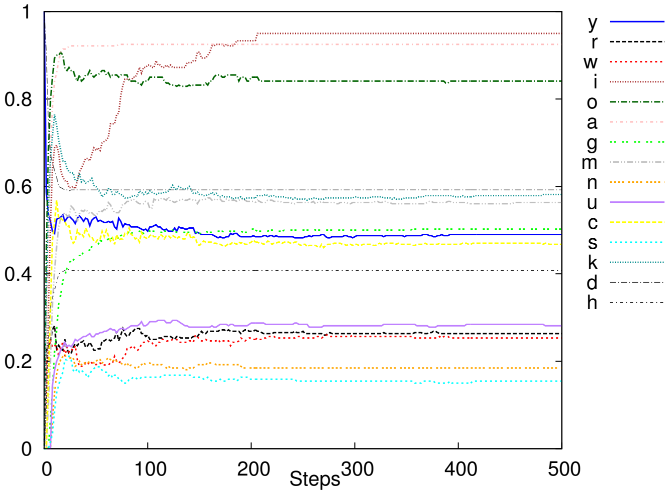
| | | Attributes ($P_1$) | | | Features ($P_2$) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| C1 | C2 | price | weight | load | $y$ | $r$ | $\cdots$ | $g$ | $\cdots$ | $d$ | $h$ |
| 600 | 15 | 391.91 | 7.80 | 33.50 | 0.57 | 0.24 | $\cdots$ | 0.0 | $\cdots$ | 0.61 | 0.39 |
| 800 | 20 | 509.83 | 11.98 | 34.45 | 0.54 | 0.23 | $\cdots$ | 0.40 | $\cdots$ | 0.60 | 0.40 |

- Probability of installing an engine ($g$) is very low, estimated at 0
(i.e. with probability 0.9 it belongs to $[0, 0.05]$, according to the confidence interval)
- This is likely because original C1 + C2 (first row) are too strict...
(average price and weight is 391.91 € and 7.8 kg, but engine costs 300 € and weighs 10 kg)

# Parametric quantitative analysis



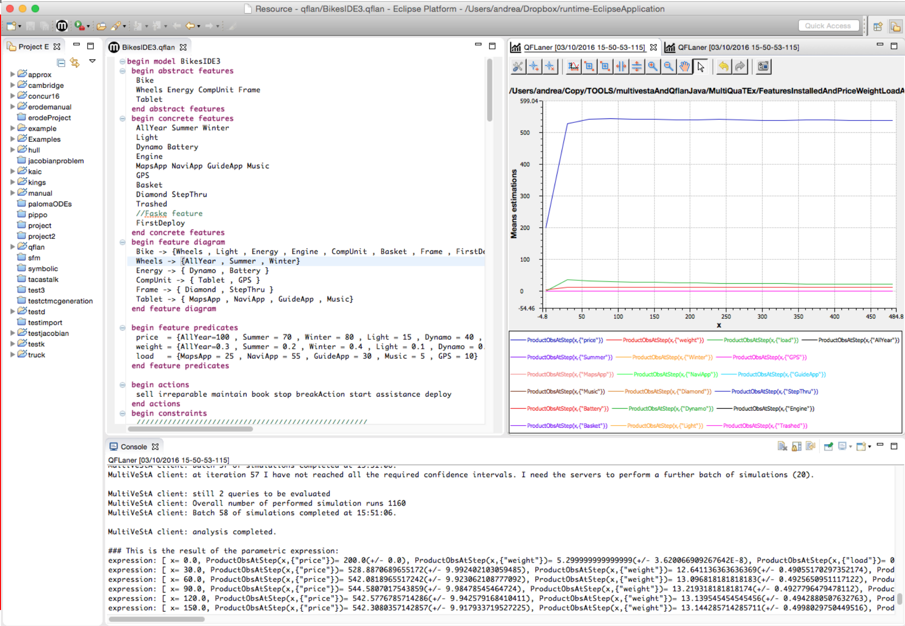
$P_2$ For each feature, the probability to be installed over time:




① initial configuration ($P(y) = P(d) = 1$), ② pre-configuration (FACTORY), ③ customization (DEPOT)

# Implementation as Eclipse-based tool

quanticol
www.quanticol.eu



## Publications acknowledging PisaMo/BicinCittà
[★: basis for this talk]




- ★ M. H. ter Beek, A. Legay, A. Lluch Lafuente, and A. Vandin, **A Quantitative Approach to Model and Analyze Dynamic Software Product Lines Using Constraints and Statistical Model Checking**. In preparation, 2017.
- M.H. ter Beek, A. Fantechi, S. Gnesi, and L. Semini, **Variability-Based Design of Services for Smart Transportation Systems**. In *Proceedings of the 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications (ISoLA'16)*. LNCS 9953, Springer, 2016, 465–481.
- ★ M.H. ter Beek, A. Legay, A. Vandin, and A. Lluch Lafuente, **Statistical Model Checking for Product Lines**. In *Proceedings of the 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques (ISoLA'16)*. LNCS 9952, Springer, 2016, 114–133.
- ★ M.H. ter Beek, S. Gnesi, D. Latella, and M. Massink, **Towards Automatic Decision Support for Bike-Sharing System Design**. In *Software Engineering and Formal Methods - Revised Selected Papers of the SEFM 2015 Collocated Workshops: ATSE, HOFM, MoKMaSD, and VERY*SCART*. LNCS 9509, Springer, 2015, 266–280.
- M.H. ter Beek, A. Fantechi, and S. Gnesi. **Applying the Product Lines Paradigm to the Quantitative Analysis of Collective Adaptive Systems**. In *Proceedings of the 19th International Software Product Line Conference (SPLC'15)*. ACM, 2015, 321–326.

- ★ M.H. ter Beek, A. Legay, A. Lluch Lafuente, and A. Vandin, **Statistical Analysis of Probabilistic Models of Software Product Lines with Quantitative Constraints.** In *Proceedings of the 19th International Software Product Line Conference (SPLC'15)*. ACM, 2015, 11–15.
- M.H. ter Beek, A. Fantechi, and S. Gnesi, **Challenges in Modelling and Analyzing Quantitative Aspects of Bike-Sharing Systems** In *Proceedings of the 6th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Technologies for Mastering Change (ISoLA'14)*. LNCS 8802, Springer, 2014, 351–367.
- M.H. ter Beek, S. Gnesi, and F. Mazzanti, **Model Checking Value-Passing Modal Specifications.** In *Perspectives of System Informatics - Revised selected papers of the 9th International Ershov Informatics Conference (PSI'14)*. LNCS 8974, Springer, 2015, 304–319.
- M.H. ter Beek, L. Bortolussi, V. Ciancia, S. Gnesi, J. Hillston, D. Latella, and M. Massink, **A Quantitative Approach to the Design and Analysis of Collective Adaptive Systems for Smart Cities**. ERCIM News - Special: Smart Cities 98 (2014), 32.

# QFLAN specification

quanticol
www.quanticol.eu

$$
\begin{aligned}
FR &\doteq [\,S \mid F\,]\\
S &\doteq DS\ \ PS\ \ QS\ \ AS\ \ IS\\
DS &\doteq \ldots \quad PS \doteq \ \ldots \quad QS \doteq \ \ldots \quad AS \doteq \ \ldots \quad IS \doteq \ \ldots\\
F &\doteq (sell, 7).D \qquad \textit{//Installing optional features:}\\
&\quad + (\mathrm{install}(s), 6).F + (\mathrm{install}(m), 10).F + (\mathrm{install}(n), 6).F + (\mathrm{install}(u), 3).F + (\mathrm{install}(c), 20).F\\
&\quad + (\mathrm{install}(g), 4).F + (\mathrm{install}(a), 5).F + (\mathrm{install}(o), 10).F + (\mathrm{install}(i), 10).F + (\mathrm{install}(k), 8).F\\
&\qquad \textit{//Replacing mandatory and exclusive features:}\\
&\quad + (\mathrm{replace}(y, r), 5).F + (\mathrm{replace}(y, w), 5).F + (\mathrm{replace}(r, y), 10).F + (\mathrm{replace}(r, w), 5).F\\
&\quad + (\mathrm{replace}(w, y), 10).F + (\mathrm{replace}(w, r), 5).F + (\mathrm{replace}(d, h), 3).F + (\mathrm{replace}(h, d), 3).F\\
D &\doteq (deploy, 10).P\\
&\qquad \textit{//Installing optional features:}\\
&\quad + \ldots \textit{same as } F\\
&\qquad \textit{//Uninstalling optional features:}\\
&\quad + \ldots \textit{same features and rates as installing, except for:}\\
&\quad + (\mathrm{uninstall}(g), 1).D + (\mathrm{uninstall}(a), 2).D + (\mathrm{uninstall}(o), 3).D\\
&\qquad \textit{//Replacing mandatory and exclusive features:}\\
&\quad + \ldots \textit{same as F, but Frame cannot be changed}\\
&\qquad \textit{//Replacing battery by dynamo in case no features requiring a Battery are installed:}\\
&\quad + (\mathrm{replace}(a, o), 1).D\\
P &\doteq (book, 10).M + (maintain, 1).D\\
M &\doteq (stop, 5).H + (break, 1).B + (c, 20).M + (i, 20).M\\
H &\doteq (start, 5).M + (break, 1).B + (park, 5).P + (c, 20).H + (i, 10).H + (s, 10).H + (u, 10).H\\
&\qquad + (m, 10).H + (n, 10).H\\
B &\doteq (assistance, 10).D + (irreparable, 1).T\\
T &\doteq (\mathrm{install}(trashed), 1).\emptyset
\end{aligned}
$$

## Constraints in the specification



$FR \doteq [\, S \mid F \,]$
$S \doteq DS\ PS\ QS\ AS\ IS$
$DS \doteq \dots \quad PS \doteq \dots \quad QS \doteq \dots \quad AS \doteq \dots \quad IS \doteq \dots$

**$FR$ is composed of process $F$ and store $S$ of constraint sets:**

$DS$ Constraints from feature diagram (incl. cross-tree constraints)

$DS \doteq y \otimes r \otimes w \quad d \otimes h \ \dots \ g \rhd a \quad n \rhd m \ \dots$

$PS$ Predicates for attributes of concrete features in feature diagram

$PS \doteq \mathrm{price}(y) = 100 \quad \mathrm{weight}(y) = 0.3 \ \dots \ \mathrm{price}(c) = 100 \quad \mathrm{load}(c) = 5 \ \dots$

$QS$ Quantitative constraints

$QS \doteq \mathrm{price}(b) \leq 800 \quad \mathrm{weight}(b) \leq 20 \quad \mathrm{load}(b) \leq 100$

$AS$ Action constraints

$AS \doteq do(sell) \rightarrow (\mathrm{price}(b) \geq 250) \ \dots \ do(c) \rightarrow has(c) \ \dots$

$IS$ Initially installed features, i.e. AllYear Wheels + Diamond Frame

$IS \doteq has(y) \quad has(d)$

# A process in the specification



$$
\begin{aligned}
FR &\doteq [\,S \mid F\,]\\
S &\doteq DS\ \ PS\ \ QS\ \ AS\ \ IS\\
DS &\doteq \ldots \quad PS \doteq \ \ldots \quad QS \doteq \ \ldots \quad AS \doteq \ \ldots \quad IS \doteq \ \ldots\\
F &\doteq (\mathit{sell}, 7).D \qquad \textit{//Installing optional features:}\\
&\quad + (\mathrm{install}(s), 6).F + (\mathrm{install}(m), 10).F + (\mathrm{install}(n), 6).F + (\mathrm{install}(u), 3).F + (\mathrm{install}(c), 20).F\\
&\quad + (\mathrm{install}(g), 4).F + (\mathrm{install}(a), 5).F + (\mathrm{install}(o), 10).F + (\mathrm{install}(i), 10).F + (\mathrm{install}(k), 8).F\\
&\qquad \textit{//Replacing mandatory and exclusive features:}\\
&\quad + (\mathrm{replace}(y, r), 5).F + (\mathrm{replace}(y, w), 5).F + (\mathrm{replace}(r, y), 10).F + (\mathrm{replace}(r, w), 5).F\\
&\quad + (\mathrm{replace}(w, y), 10).F + (\mathrm{replace}(w, r), 5).F + (\mathrm{replace}(d, h), 3).F + (\mathrm{replace}(h, d), 3).F
\end{aligned}
$$

**$F$ implements FACTORY's behavior as a *weighted* choice among:**

(1) With rate 7, the bike is sold and sent to depot. This action can only be executed if C4: $do(\mathit{sell}) \rightarrow (\mathrm{price}(b) \geq 250)$ is respected

(2) Install optional features and iterate on $F$. Such installations are only performed if $DS$ and $QS$ are preserved

(3) Replace pre-installed mandatory exclusive features $IS$, i.e. Frame or Wheels. Again, $DS$ and $QS$ must be preserved

QFLAN's semantics forbids re-installing (installed) features